오픈소스 거버넌스 자동화와 소프트웨어 공급망 관리

OSC Korea | Sonatype 한국 총판

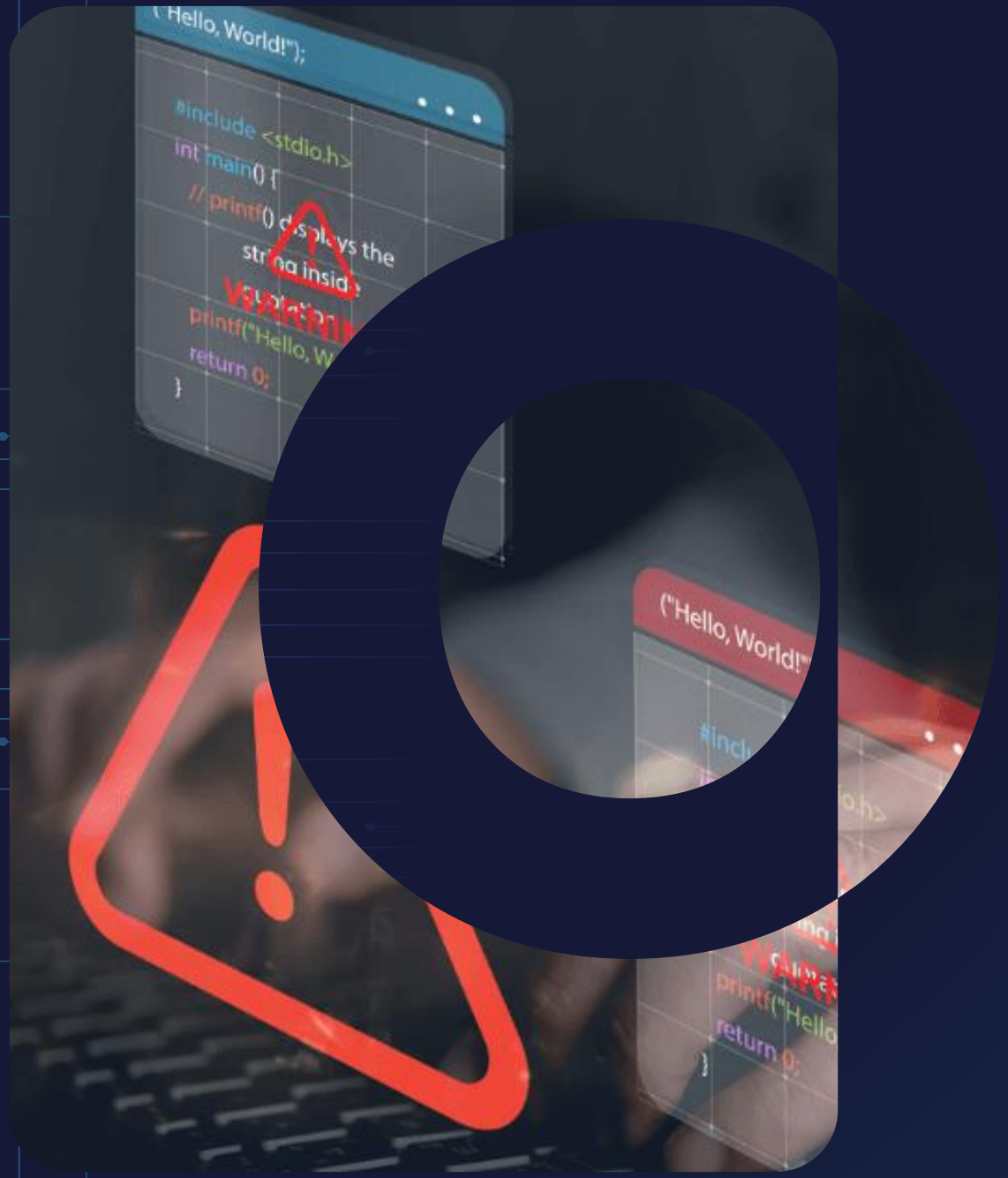
750k
매주 릴리즈 되는 신규 오픈소스

100k
매주 식별되는 새로운 보안위협

2,689
관리대상 오픈소스 라이선스 종류

27 Days
NVD에서 CVE 등록에 소요되는 시간

목차



Software Supply Chain Challenges & Attacks

- 소프트웨어 공급망 관리의 도전과제 및 공격 기법
- 오픈소스 거버넌스 자동화 고려사항
- 오픈소스 거버넌스 자동화 구현 방안
- SBOM 통합관리를 위한 SBOM Manager

오픈소스 거버넌스 영역의 전개

2000's : 라이선스

각종 라이선스 분쟁

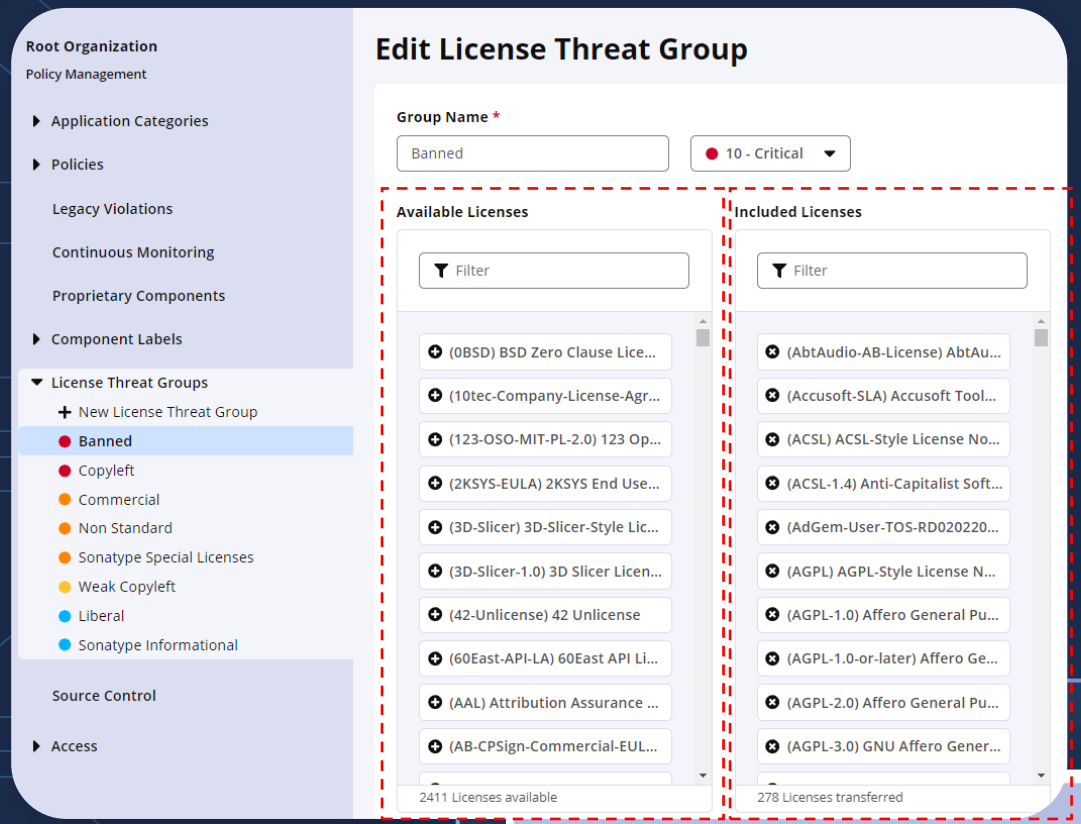
2010's : 취약점

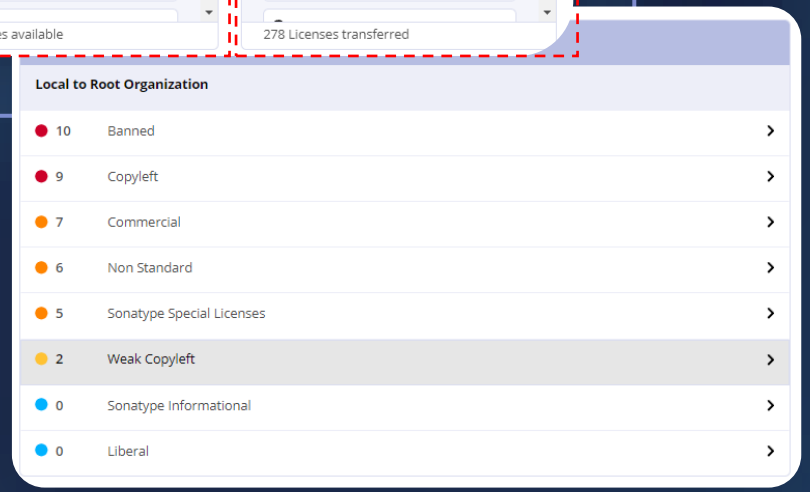
취약점을 이용한 대형 보안침해 사고

2020's : 멀웨어

공급망을 통한 악성코드/멀웨어 공격패턴 급증

라이선스 관리





관리대상 라이선스 종류 : 2,689 종 (24년 8월)

❖라이선스 관리의 복잡성

- 오픈소스 **도입 초기**에는 관리할 의존성이 많지 않았고, 라이선스 종류도 많지 않았음 (MIT, BSD, GPL/AGPL, Mozilla)
- 프로젝트당 **평균 +150여개의 오픈소스**를 사용하고 라이선스의 종류도 지속적으로 많아지고 있음
- **라이선스별 요구사항**을 파악하고 **고지의무 등을 준수**하기 위해 자동화된 도구의 사용이 필요함

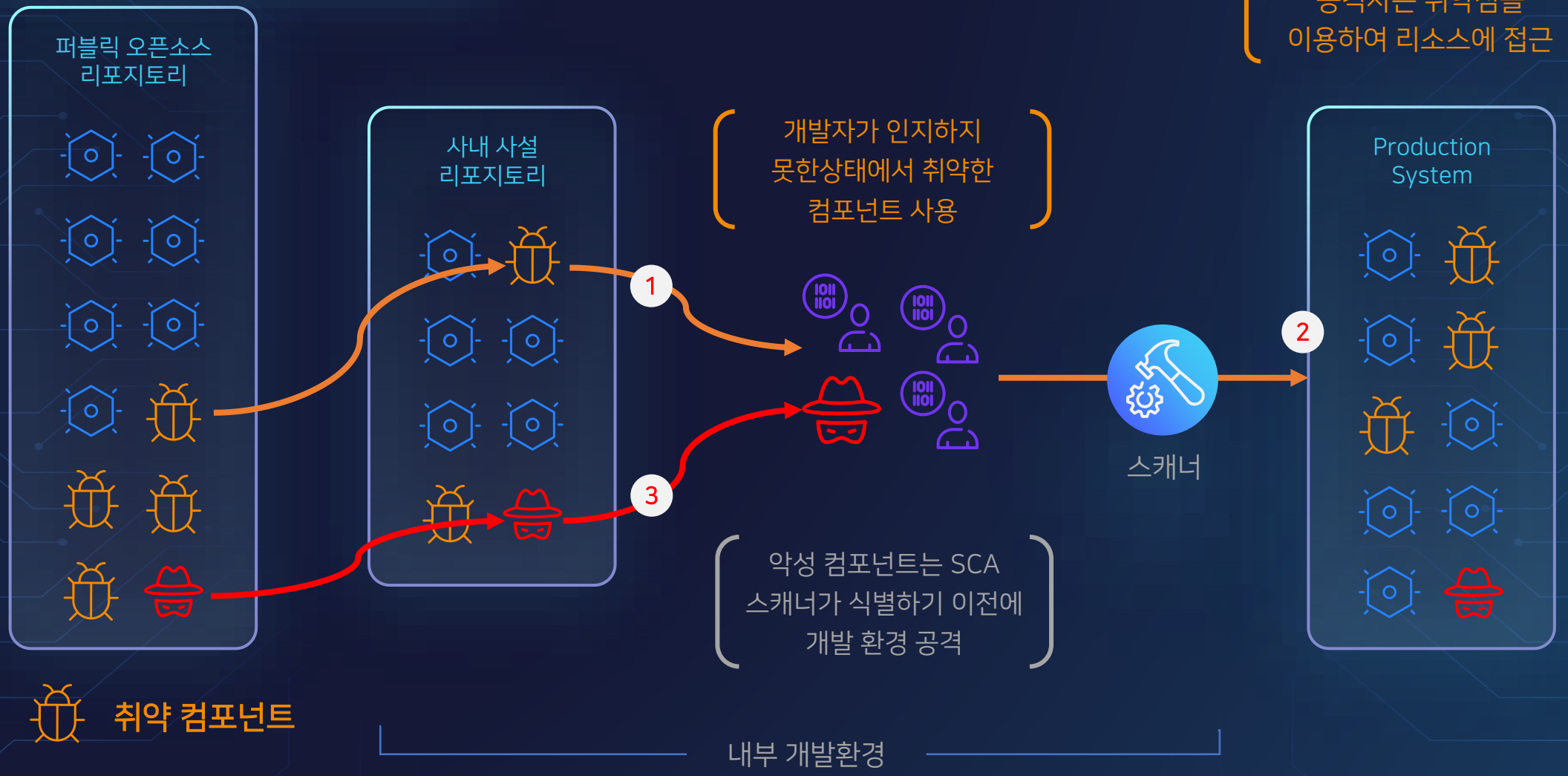
Banned	Copyleft	Commercial	Non Standard	Weak Copyleft	Liberal
278	61	927	131	202	780

AGPL Free as in Freedom · GPL Free as in Freedom · NVIDIA CUDA · BEERWARE · mozilla · BSD 6

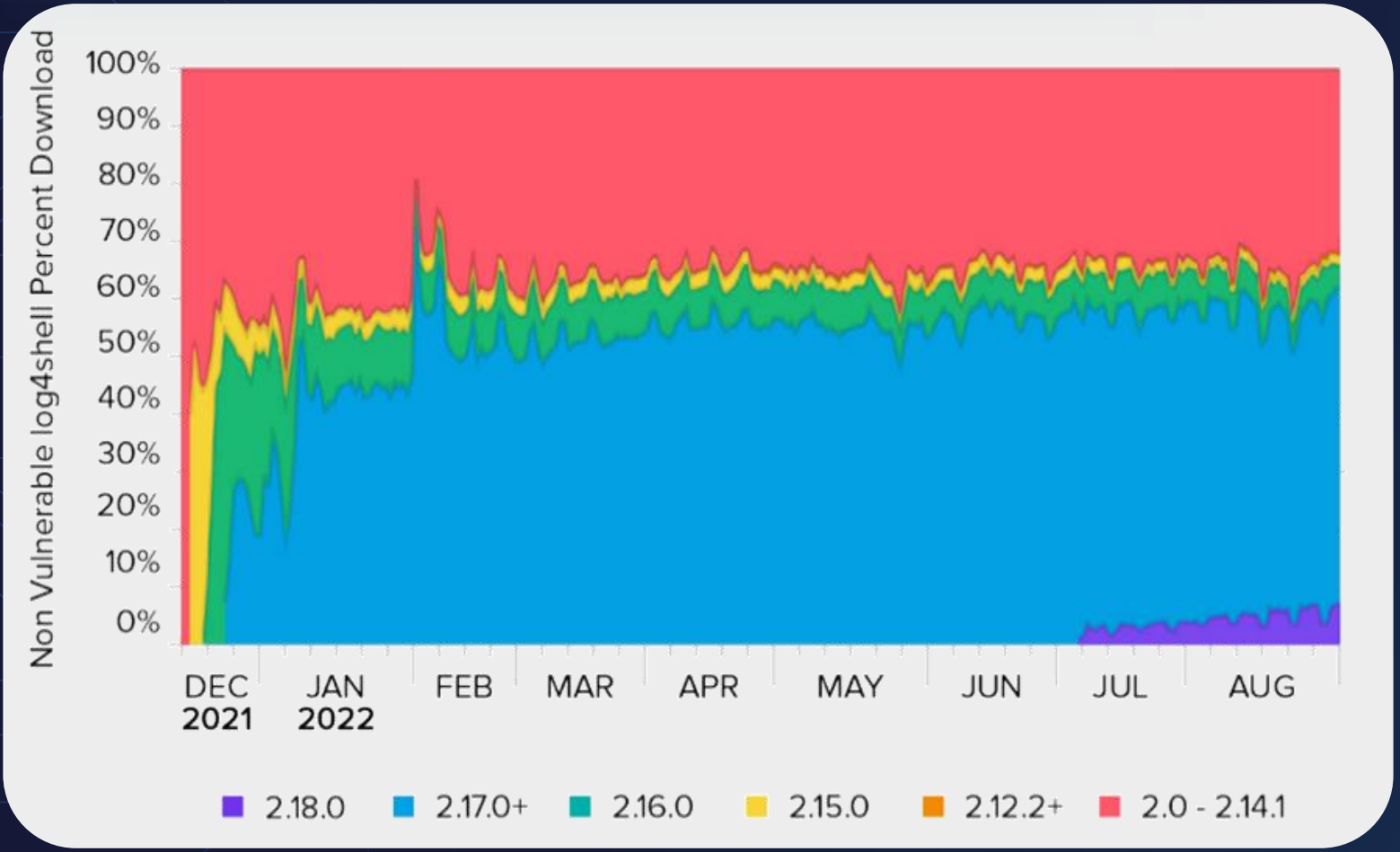
소프트웨어 공급망을 통한 보안 위협

01. Vulnerability (취약점)	02. Malware (악성코드)
악용(Exploited)될 수 있는 결함으로 인해 취약한 컴포넌트	유해한 코드를 주입하기 위해 의도적으로 만들어진 악성 컴포넌트
CVE 로 관리	CVE로 관리되지 않음
침해 행위가 일어나는 경우 피해	내부로 유입되면 즉각적인 피해

개발환경의 위협요소


퍼블릭 오픈소스
리포지토리
사내 사설
리포지토리
개발자가 인지하지
못한상태에서 취약한
컴포넌트 사용
공격자는 취약점을
이용하여 리소스에 접근
Production
System
1
2
3
스캐너
악성 컴포넌트는 SCA
스캐너가 식별하기 이전에
개발 환경 공격
취약 컴포넌트
악성 컴포넌트
내부 개발환경

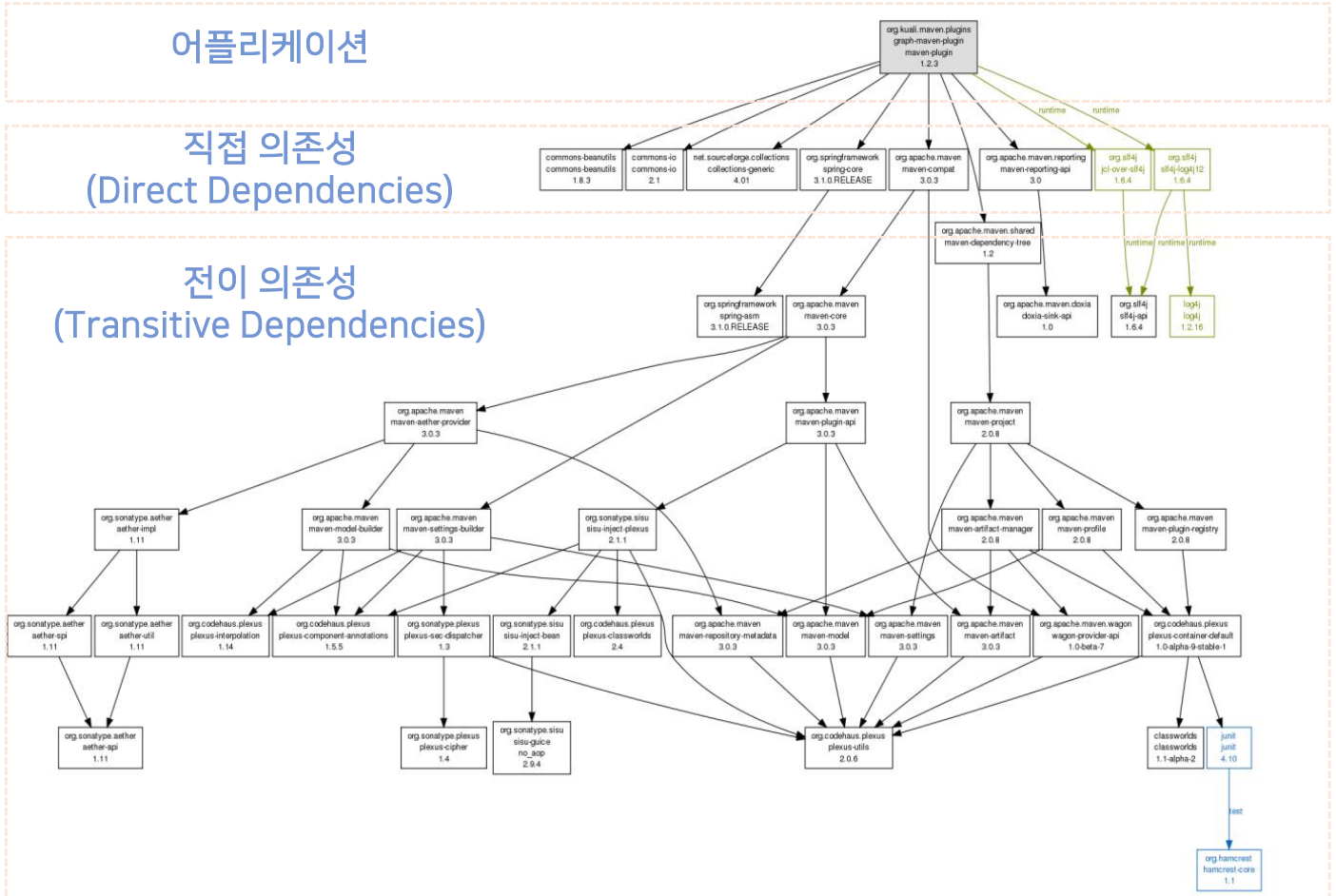
취약 컴포넌트 다운로드 현황



2021년 Log4j 취약점이 공개된 후 개선된 버전이 다수 릴리즈 되었지만 여전히 취약한 버전의 다운로드 비중이 30%를 차지함

Source: Log4Shell Vulnerable Downloads - Sonatype (Maven Central)

오픈소스의 의존성(Dependency)



- 어플리케이션당 평균 +150개의 의존성을 가짐
- 현대화된 개발방법론에서는 패키지 매니저를 통해 복잡한 의존성을 관리함
- 전이 의존성은 직접 의존성이 참조하는 독립된 프로젝트
- 전이 의존성은 다른 프로젝트에서는 직접 의존성으로 사용될 수 있음
- 전이 의존성은 개발자가 직접 참조하지 않기 때문에 사용여부를 인지하지 못할 수 있음

소프트웨어 공급망 공격 기법 #1

타이포스쿼팅

- 주요 패키지명의 **타이핑오류**를 활용하는 기법
- 정상 패키지와 **비슷하게 보이는 악성 패키지**를 만든 후, NPM Repository등에 업로드
- 개발자들이 의존성을 정의할 때 이름을 잘못 입력하는 경우, 의도된 **악성 패키지가 다운로드 되어 공격**에 이용되는 방식
- 배치 스크립트는 윈도우의 레지스트리를 변경하거나, 트로이목마 또는 랜섬웨어를 통해 대상 호스트를 감염시킴

1 타이포스쿼팅 예시

1. cross-env -> crossenv
2. express -> exprss
3. electron -> electorn

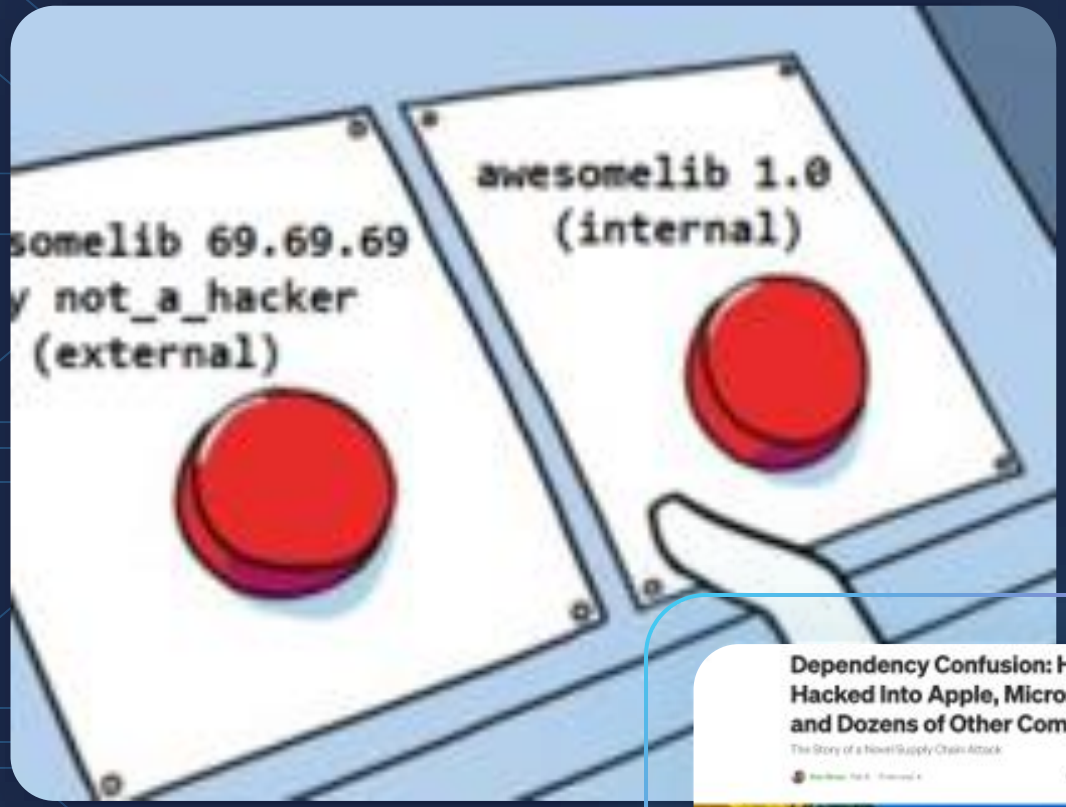
2 공격 형태

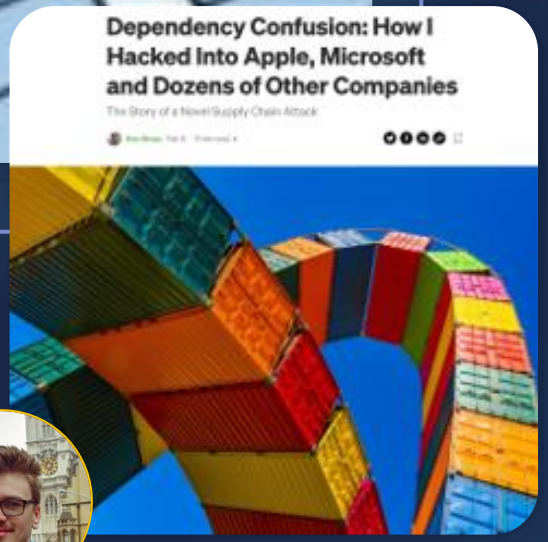
1. Data exfiltration (데이터 탈취)
2. Code Injection (코드 삽입)
3. Crypto Stealer / Miner (암호탈취 / 마이닝)
4. Denial of Service (서비스 거부공격)
5. Obfuscated Code (난독화 코드)
6. Backdoor Install (백도어 설치)
7. File System Corruption (파일시스템 손상)

3 최근 사례 (2024년 8월)

1. 정식 프로젝트 (블럭체인 지원 Python APIs)
   - Github : solana-py v0.34.3
   - PyPI : solana
2. 악성 코드 배포
   - PyPI : solana-py v0.34.5
   - 전자지갑 암호 키 탈취

출처 : https://www.sonatype.com/blog/an-ideal-pypi-typosquat-solana-py-is-here-to-steal-your-crypto-keys





Dependency Confusion
(의존성 혼동)

❖ Alex Birsan (2021년)

- 공개 저장소의 보안강화 이후 다른 형태의 Supply Chain 공격 방식 등장
- 내부 어플리케이션에서 사용하는 패키지명을 찾아낸 후 **내부의 낮은 버전보다 외부 최신 Dependency를 우선하는 패키지 매니저의 특성을** 활용한 기법
- 피해기업 : Apple, Microsoft, Netflix, PayPal, Shopify, Tesla and Uber 회사 등

❖ PyTorch (2022년)

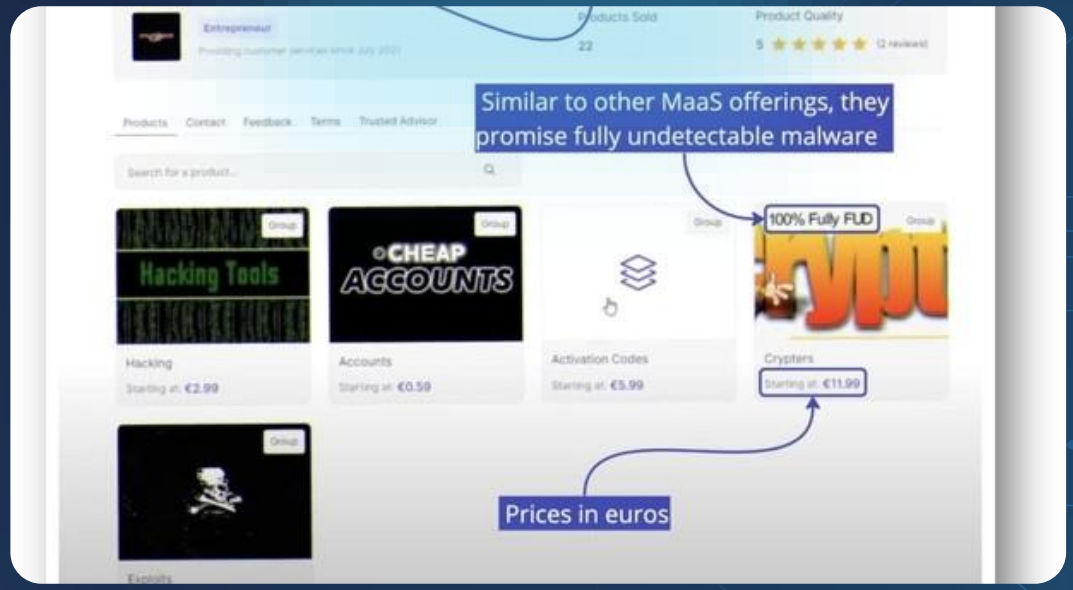
- PyTouch-nightly 빌드가 **사설 저장소에서 패키지 (torchtriton)를 다운**받는 허점을 이용해 공식 저장소에 동명의 악성 패키지를 높은 버전으로 업로드 함
- **Python 공식 저장소(pypi.org)가 사설 저장소보다 우선**하여 악성 패키지가 일반 사용자에게 배포됨

출처 : https://www.sonatype.com/blog/pytorch-namespace-dependency-confusion-attack

Malicious Code Injection (악성코드 주입)

Malicious Code Injection

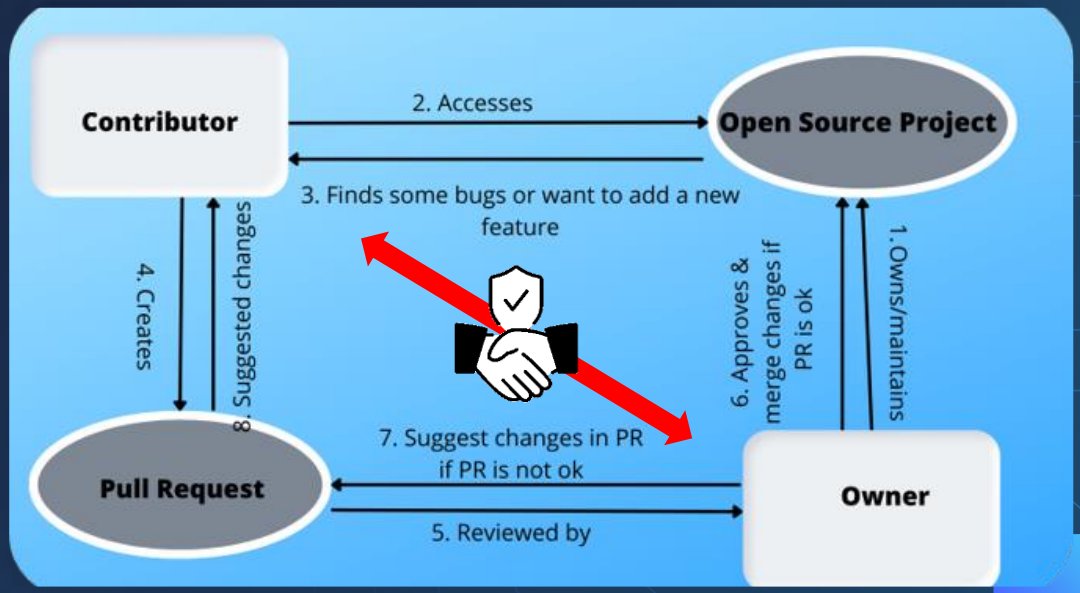
Malware as a Service



출처 : https://blog.sonatype.com/malware-monthly-march-2023

정상 프로젝트를 통한 잠입

"활동이 적은 프로젝트의 **관리 허점**을 찾거나 프로젝트 Owner와 **신뢰관계를 구축**하여 Commit 권한 획득"



ws users, revives malware ...

PyPI crypto-stealer targets Windows users, revives malware campaign

May 29, 2024 By Ax Sharma

7 minute read time

FOLDERS

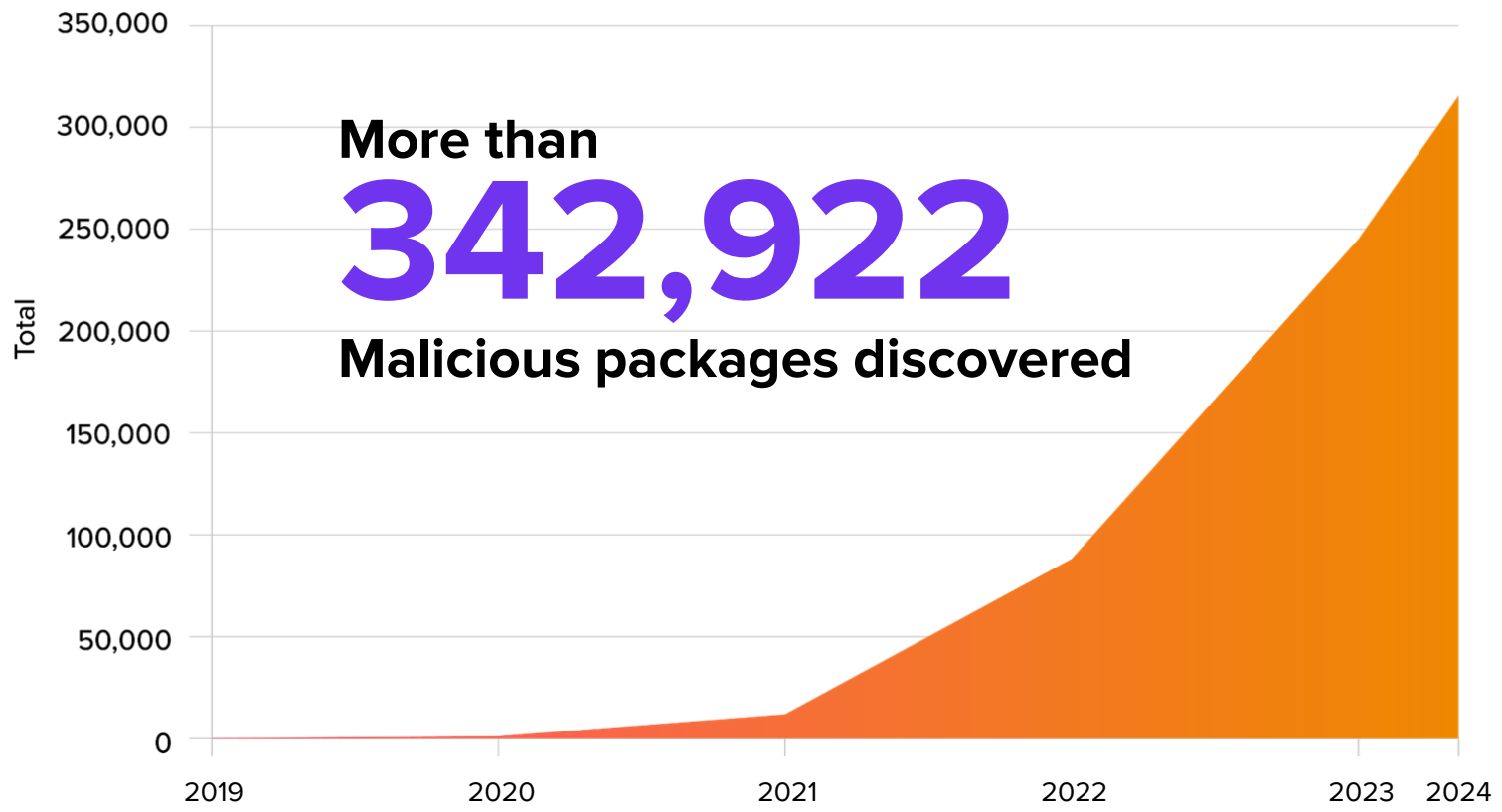
- pytoileur
  - 1.0.1
    - pytoileur-1.0.1.tar
      - pytoileur-1.0.1
        - pytoileur.egg-info
          - dependency_links.txt
          - PKG-INFO
          - SOURCES.txt
          - top_level.txt
        - PKG-INFO
        - README.md
        - setup.cfg
        - setup.py
    - pytoileur-1.0.1.tar.gz
  - 1.0.2

setup.py

```python
from setuptools import setup, find_packages
from setuptools.command.install import install
from pathlib import Path
import os


VERSION = '1.0.1'
DESCRIPTION = 'Cool package.'
this_directory = Path(__file__).parent
long_description = (this_directory / "README.md").read_text()


class InstallCommand(install):

    def run(self):
        try:
            print("")
        except:
            pass
        install.run(self)


setup(
    name="pytoileur",
    version=VERSION,
    author="HW",
    author_email="",
    description=DESCRIPTION,
    long_description_content_type="text/markdown",
    long_description=long_description,
    packages=find_packages(),
    install_requires=[],
    keywords=[],
    classifiers=[
        "Development Status :: 1 - Planning",
        "Intended Audience :: Developers",
        "Programming Language :: Python :: 3",
        "Operating System :: Unix",
        "Operating System :: MacOS :: MacOS X",
        "Operating System :: Microsoft :: Windows",
    ],
    cmdclass={
        'install': InstallCommand
    }
)
```

base64-encoded payload
실행 명령어

악성 소프트웨어를 통한 공급망 공격 현황



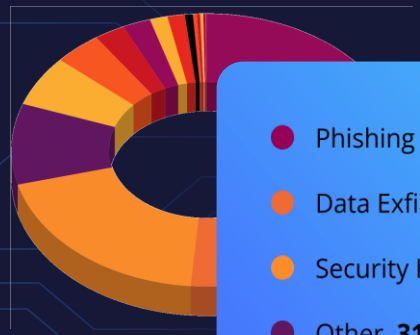
2023년에는 이전해까지 발견된
모든 악성패키지의 2배에
달하는 악성패키지가 탐지됨

출처 : Sonatype Annual State of the Software Supply Chain
https://www.sonatype.com/state-of-the-software-supply-chain/open-source-supply-and-demand

악성 소프트웨어 식별 및 대응 속도

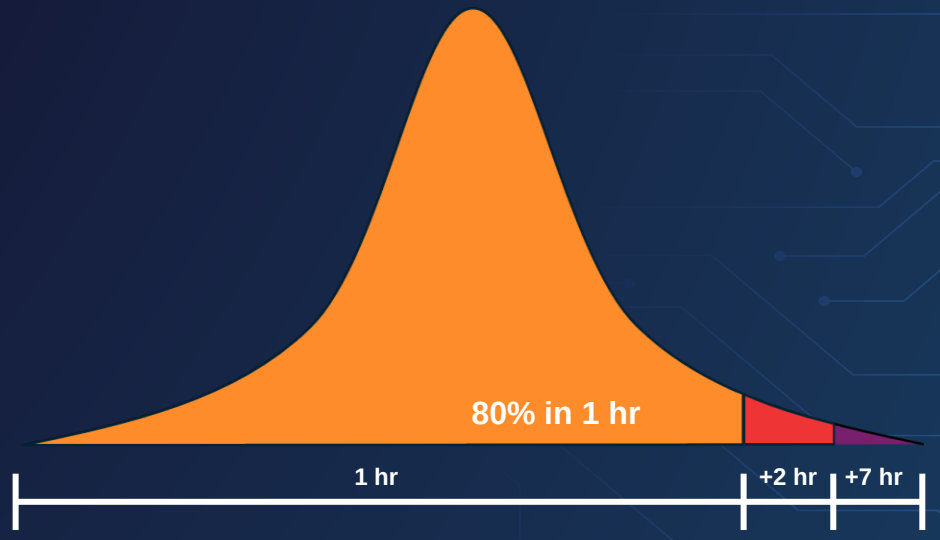
342,922
malicious versions

Sonatype Discovered **289,933**



- Phishing **96,465**
- Data Exfiltration **73,008**
- Security Holding **64,779**
- Other **31,652**
- PII Exfiltration **19,772**
- Backdoor **13,065**
- Research Project **8,421**
- PUA **7,669**
- Crypto Stealer/Miner **4,837**
- Dropper **4,734**
- File System Corruption **2,212**
- Protestware **1,067**
- Code Injection **911**
- Denial of Service **848**
- Credential Exfiltration **600**
- Obfuscated Code **204**
- Ransomware **37**
- Hijacker **19**
- URL Redirector **8**

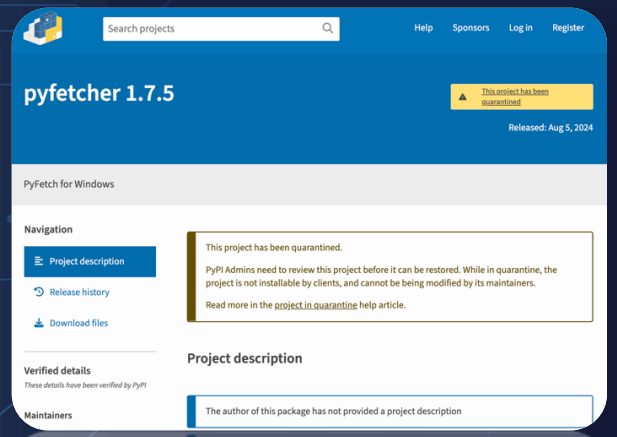
식별된 악성소프트웨어 중 80%는 1시간 내에 Sonatype Database에 추가

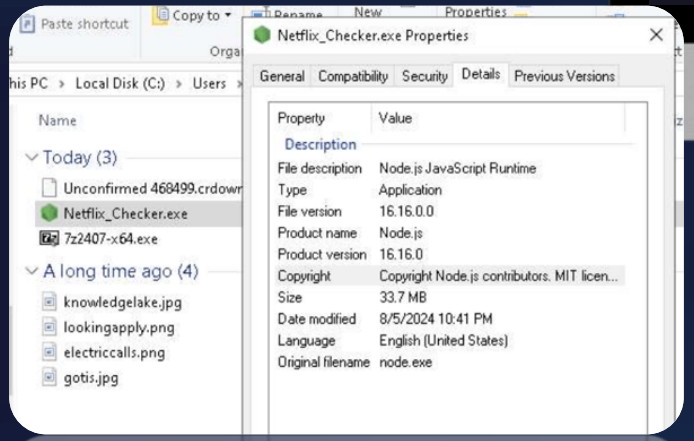


소프트웨어 공급망 vs. Anti-Virus

- Sonatype 보안팀에서 탐지한 악성 PyPI 패키지 (netfetcher, pyfetcher)
- 동명의 윈도우 유틸리티를 가장하여 특정 URL로부터 악성 실행 화일을 다운로드 후 이름 변경 (netflix_checker.exe)
- 공식 NodeJS 로고와 메타데이터 포함하여 공식 패키지로 가장
- 2024년 8월 22일 현재 C2서버 Active 상태
- Anti-Virus 도구에서 탐지 되지 않음

Anti-Virus 검색 결과 > Undetected





출처 : https://www.sonatype.com/blog/pyfetcher-netfetch-drop-netflix-checker-on-windows

Governance Automation Consideration

- 소프트웨어 공급망 관리의 도전과제 및 공격 기법
- 오픈소스 거버넌스 자동화 고려사항
- 오픈소스 거버넌스 자동화 구현 방안
- SBOM 통합관리를 위한 SBOM Manager

오픈소스 거버넌스 자동화 요건

오픈소스 거버넌스의 자동화는 고품질의 데이터를 기반으로 정확하게 오픈소스 컴포넌트를 식별하여
오탐이 없는 환경을 구축하고, 단계별 정책을 적용하여 전 방위적인 리스크 관리체계 확보를 목표로 함

Self-Remediation (셀프서비스 진단/치유) ›

고품질 데이터베이스 확보 방식

Automated Vulnerability Detection

Prioritization

Public Sources : Security Feeds & Others
- CVE Feeds
- 보안 공지사이트 (Security Advisories)
- 프로젝트 보안 공지
- FSISAC & Customer Report
- Email, Blog, OWASP, OSSIndex

Github 모든 오픈소스 Commit / Event 모니터링
- CVE, SQL Injection, XXE, RCE, XSS

60여개 시그널을 통한 AI/ML 기반 Behavior 모니터링
- Public Repository (NPM, PyPI, Maven등) 24X7 감시

Human Curation

Sonatype 보안 리서치팀 Curation (+65명)
- 소스코드 분석 (Class, Method)
- 보안이슈(Defect)의 근본 출처 확인
- 불일치(Discrepancy), 모호함(Vagueness), 부정확성 (Inaccuracy) 제거

Deep Dive 정보 추가
- Sonatype (정보 소스) 고유정보 추가
- Remediation 추가

추가 정보 수집
- Github 및 프로젝트 다운로드 사이트 분석
- 인기도, Release History, Declared License

Expansion

Secondary Expansion

추가 분석 및 취약점간 상호 연계
- 신규 취약점에 대해 기존 다른 컴포넌트에 미치는 영향 추가 분석
- 해당 취약점이 복수의 라이브러리에 영향을 미치고 있는지 확인
- Secondary Expansion 분석을 통해 현재까지 3,800만개의 취약점 정보 추가

오탐 방지 효과
- False Negative : 0.1%
- False Positive : 0.6%

XXE : XML External Entity, RCE : Remote Code Execution, XSS : Cross Site Scripting

Sonatype Data Story – 주요 지표

sonatype

Breadth

270M
오픈소스 카탈로그

8M/Day
Github/보안공지

Depth

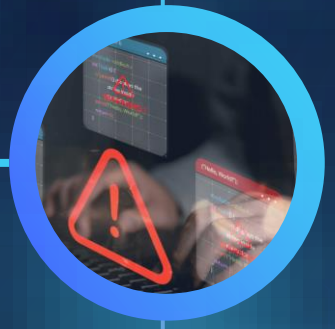
55M
취약점 데이터

14M
Deep Dive (심층분석)

Speed

10 mins
OSS 카탈로그

1 Hour
악성패키지지

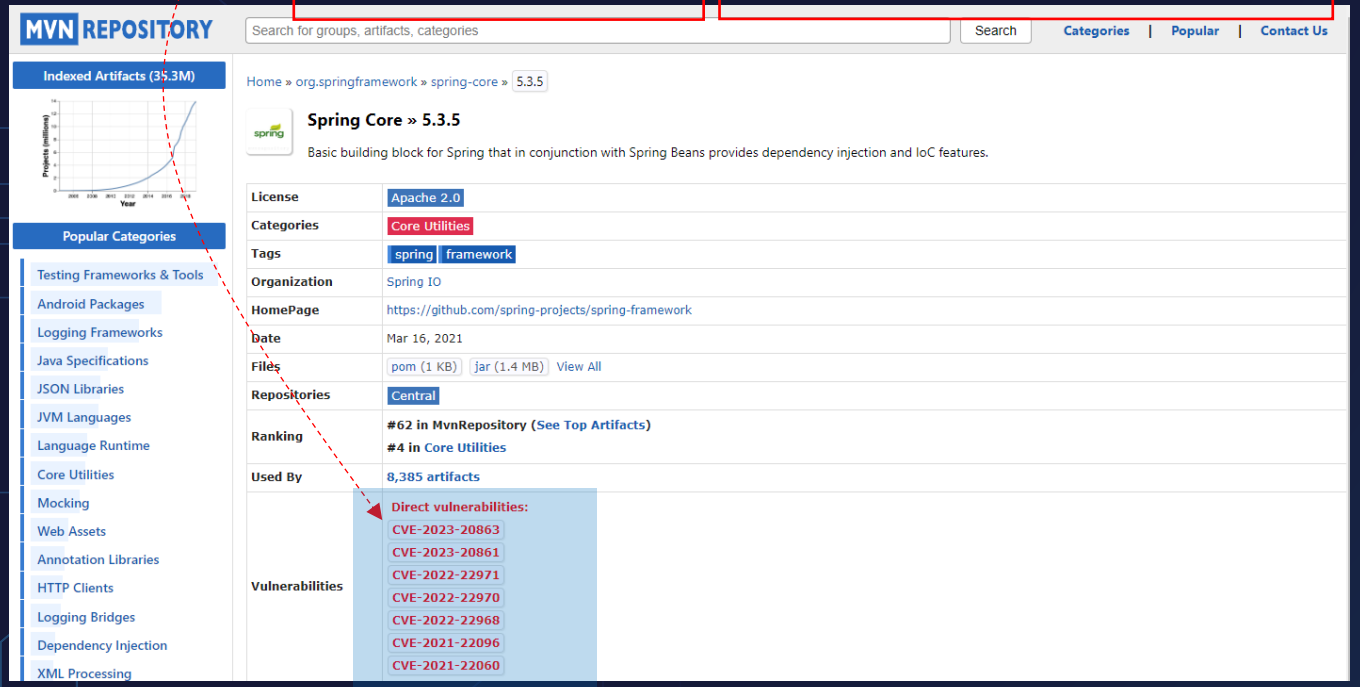


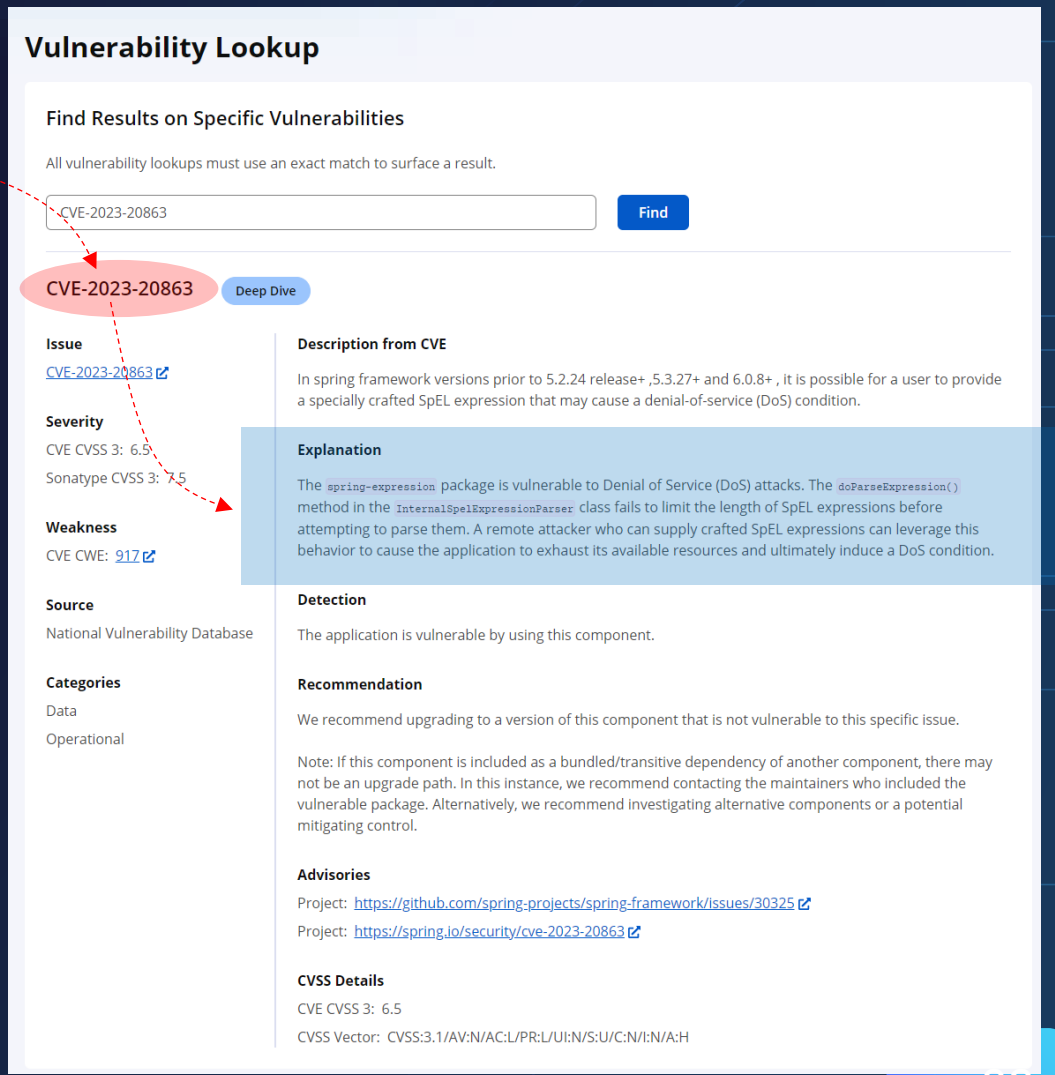
Accuracy

32%
Sonatype 에서 수정한
퍼블릭 보안공지

Sonatype Data 품질 평가 사례 #1 (False Positive)

점검대상	레퍼런스 CVEs	Sonatype 점검결과
spring-core-5.3.5.jar	CVE-2023-20863	spring-expression
	CVE-2023-20861	spring-expression
	CVE-2022-22971	spring-messaging
	CVE-2022-22970	spring-beans
	CVE-2022-22968	spring-context
	CVE-2022-22965	spring-beans
	CVE-2022-22950	spring-expression
	CVE-2021-22118	spring-web
	CVE-2021-22096	spring-web
	CVE-2021-22060	spring-webmvc, spring-webflux, spring-websocket





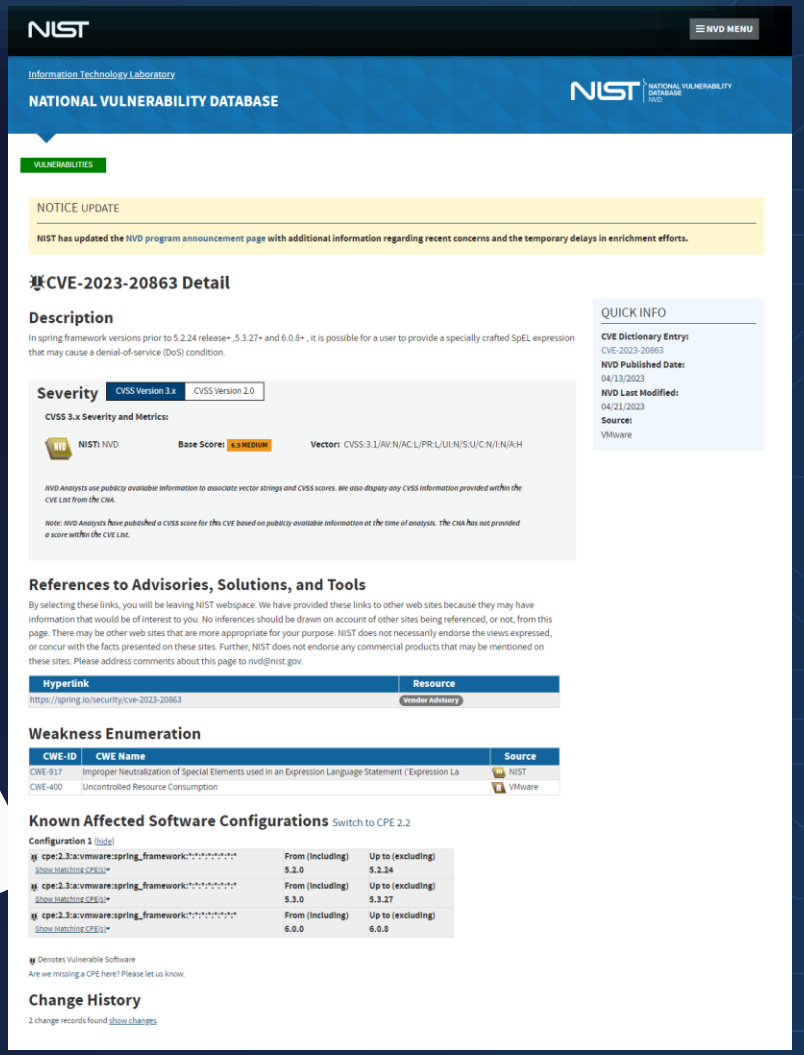
- MVN Repository 포털에는 Spring Core에 대한 CVE들이 기록되어 있으나 실제 해당 CVE들은 Spring Core가 아닌 다른 컴포넌트에 대한 CVE들입니다.
- Public 데이터에 만 의존한 진단을 수행하는 솔루션의 경우에는 오탐 (과탐 : False Positive)을 유발하게 됩니다

Sonatype Data 품질 평가 사례 #1 (False Positive) – Cont.

Known Affected Software Configurations Switch to CPE 2.2

Configuration 1 (hide)

	From (Including)	Up to (excluding)
cpe:2.3:a:vmware:spring_framework:*:*:*:*:*:*:*:*		
Show Matching CPE(s)	5.2.0	5.2.24
cpe:2.3:a:vmware:spring_framework:*:*:*:*:*:*:*:*		
Hide Matching CPE(s)	5.3.0	5.3.27
• cpe:2.3:a:vmware:spring_framework:5.3.0:-:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.0:milestone1:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.0:milestone2:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.0:rc1:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.0:rc2:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.1:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.2:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.3:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.4:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.5:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.6:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.7:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.8:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.9:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.10:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.11:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.12:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.13:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.14:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.15:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.16:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.17:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.18:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.19:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.20:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.21:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.22:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.23:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.24:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.25:*:*:*:*:*:*:*		
• cpe:2.3:a:vmware:spring_framework:5.3.26:*:*:*:*:*:*:*		
cpe:2.3:a:vmware:spring_framework:*:*:*:*:*:*:*:*		
Show Matching CPE(s)	6.0.0	6.0.8



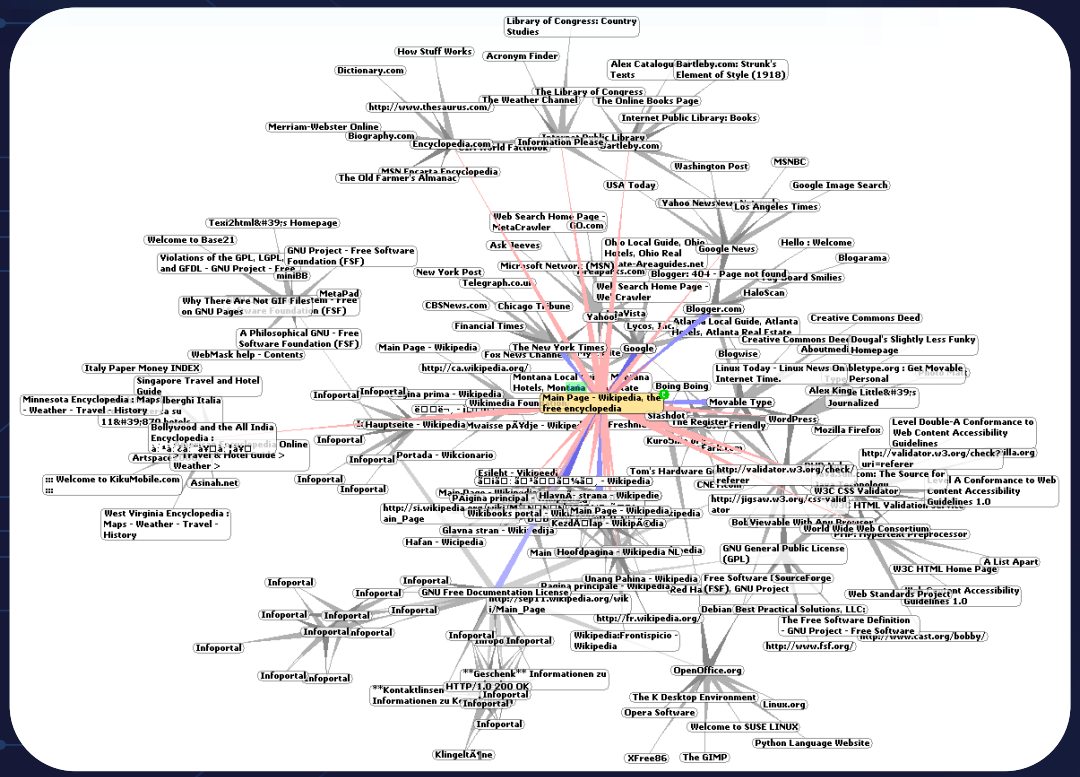
https://nvd.nist.gov/vuln/detail/CVE-2023-20863#range-9106059

- Public Data가 부정확한 배경에는 여러가지 요인이 있으나 그중 하나는 CVE 정보가 와일드카드 방식으로 맵핑되어 있기 때문입니다.
- 상위 프로젝트는 매칭하지만 하위 컴포넌트 수준에서 매칭되지 않는 결과로 이어지는 경우가 많습니다

Governance Automation Implementation

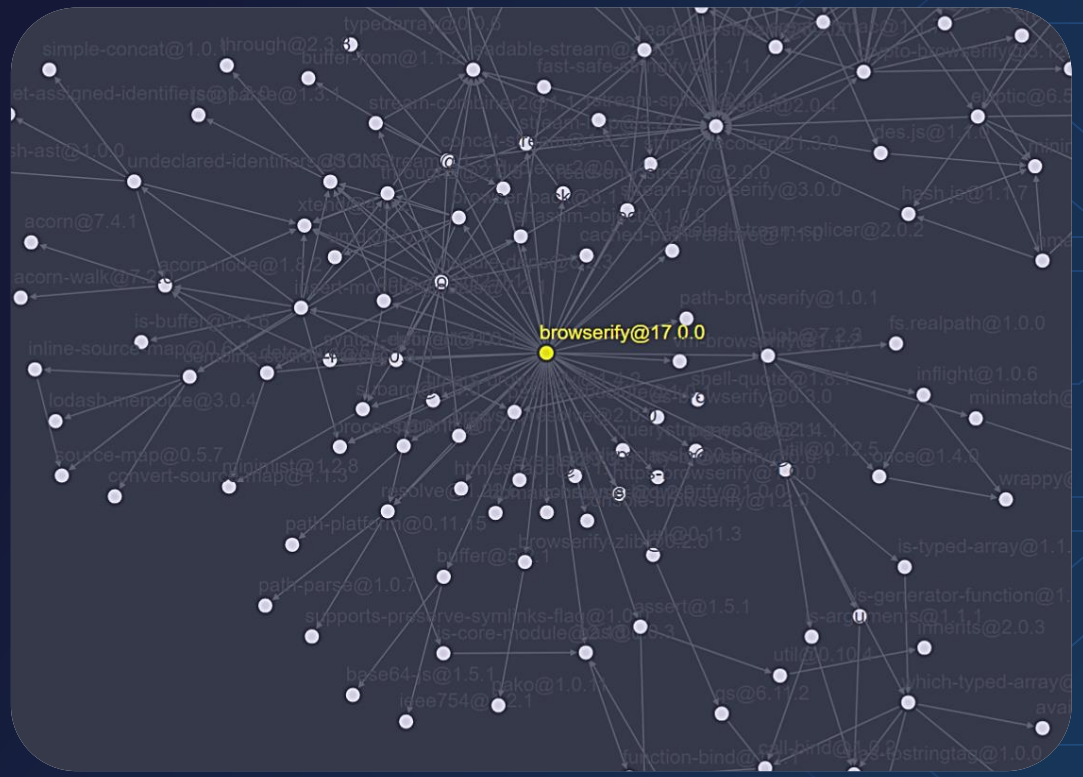
- 소프트웨어 공급망 관리의 도전과제 및 공격 기법
- 오픈소스 거버넌스 자동화 고려사항
- **오픈소스 거버넌스 자동화 구현 방안**
- SBOM 통합관리를 위한 SBOM Manager

오픈소스 거버넌스 특성



Web Hyperlink

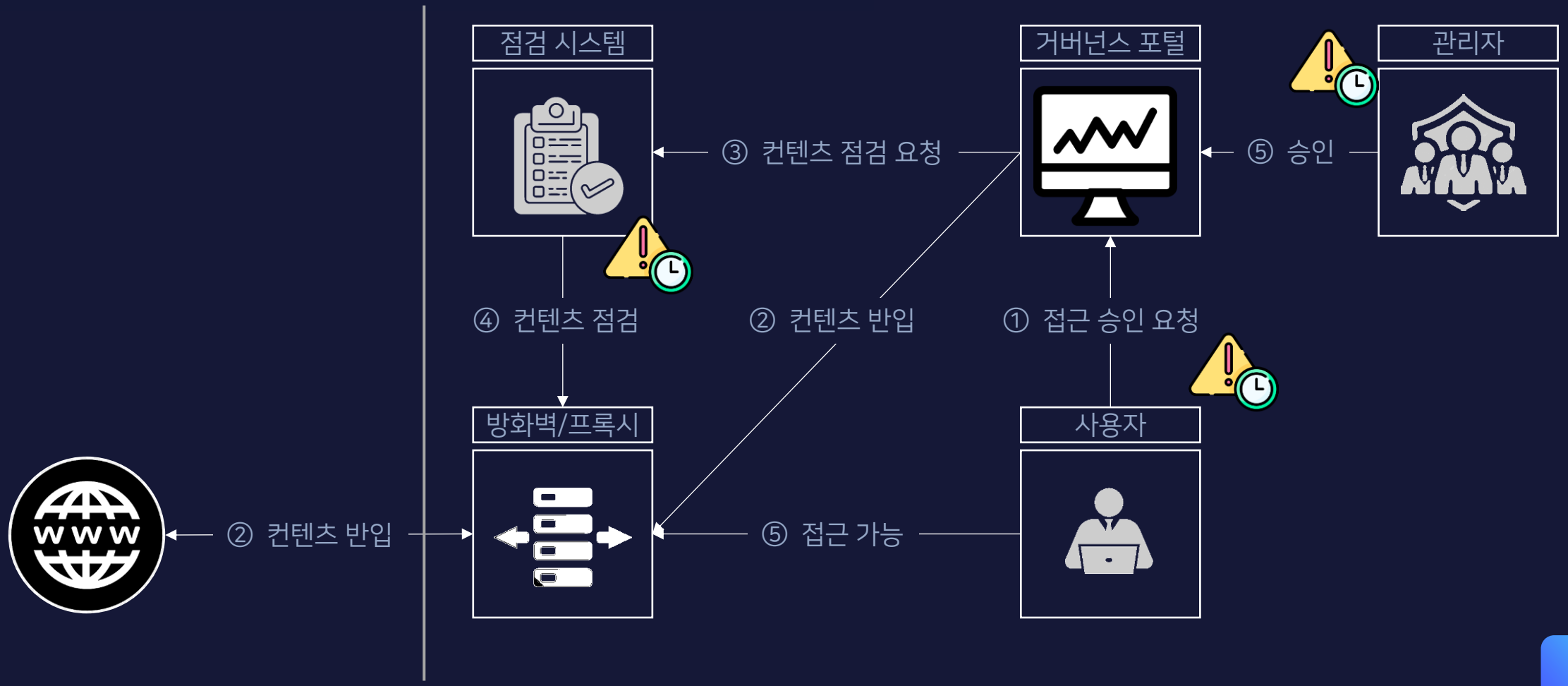
https://en.wikipedia.org/wiki/World_Wide_Web



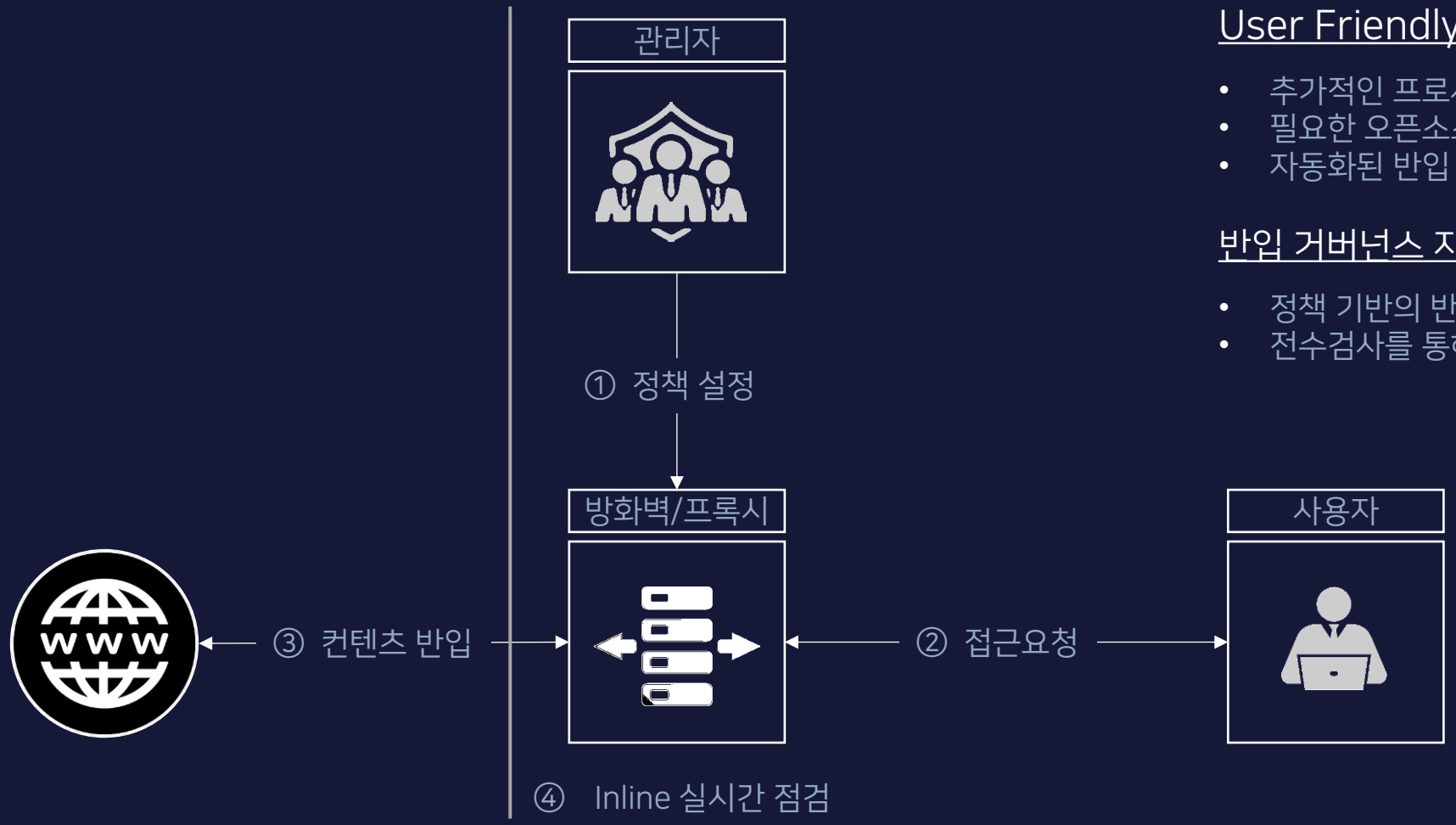
Dependency (의존성)

https://npm.anvaka.com/#/

거버넌스 구현 방식 비교 – 포털 방식



거버넌스 구현 방식 비교 – Inline 방식



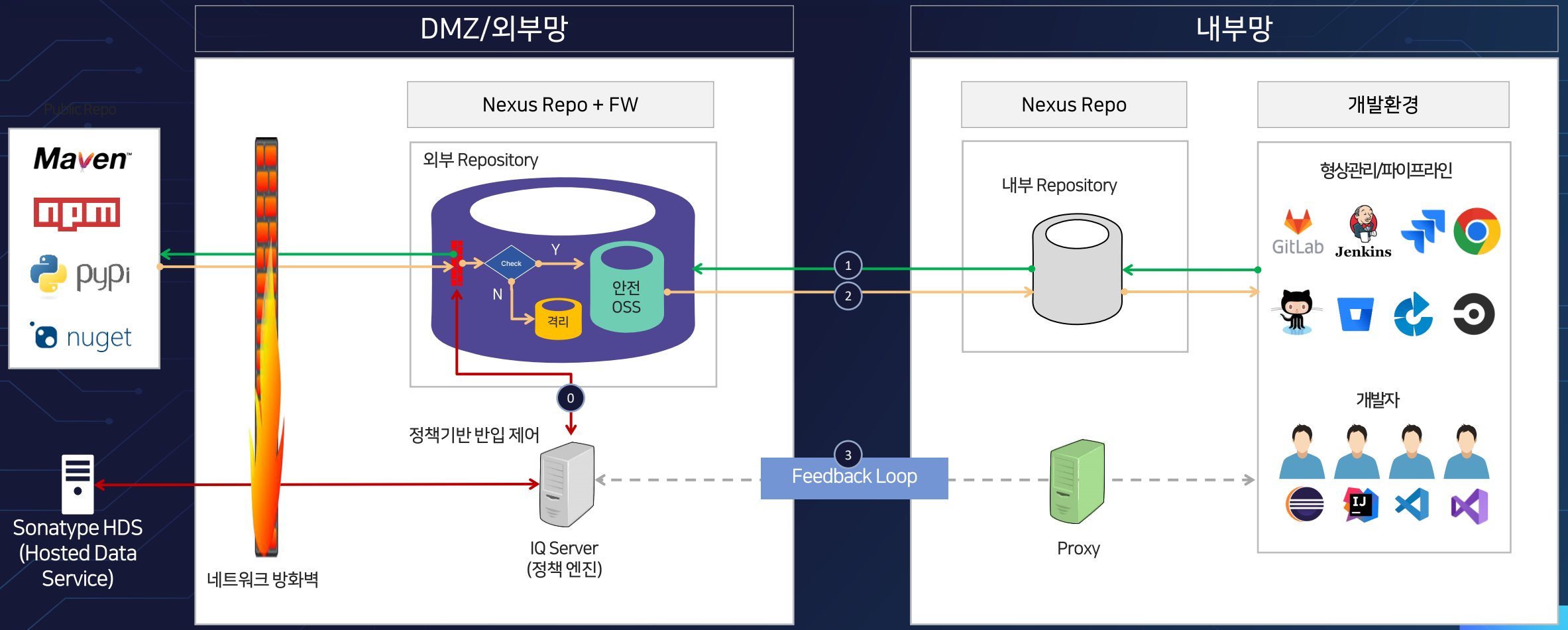
User Friendly

- 추가적인 프로세스 필요 없음
- 필요한 오픈소스를 즉시 활용하여 생산성 향상
- 자동화된 반입 프로세스로 개발시간 단축

반입 거버넌스 자동화

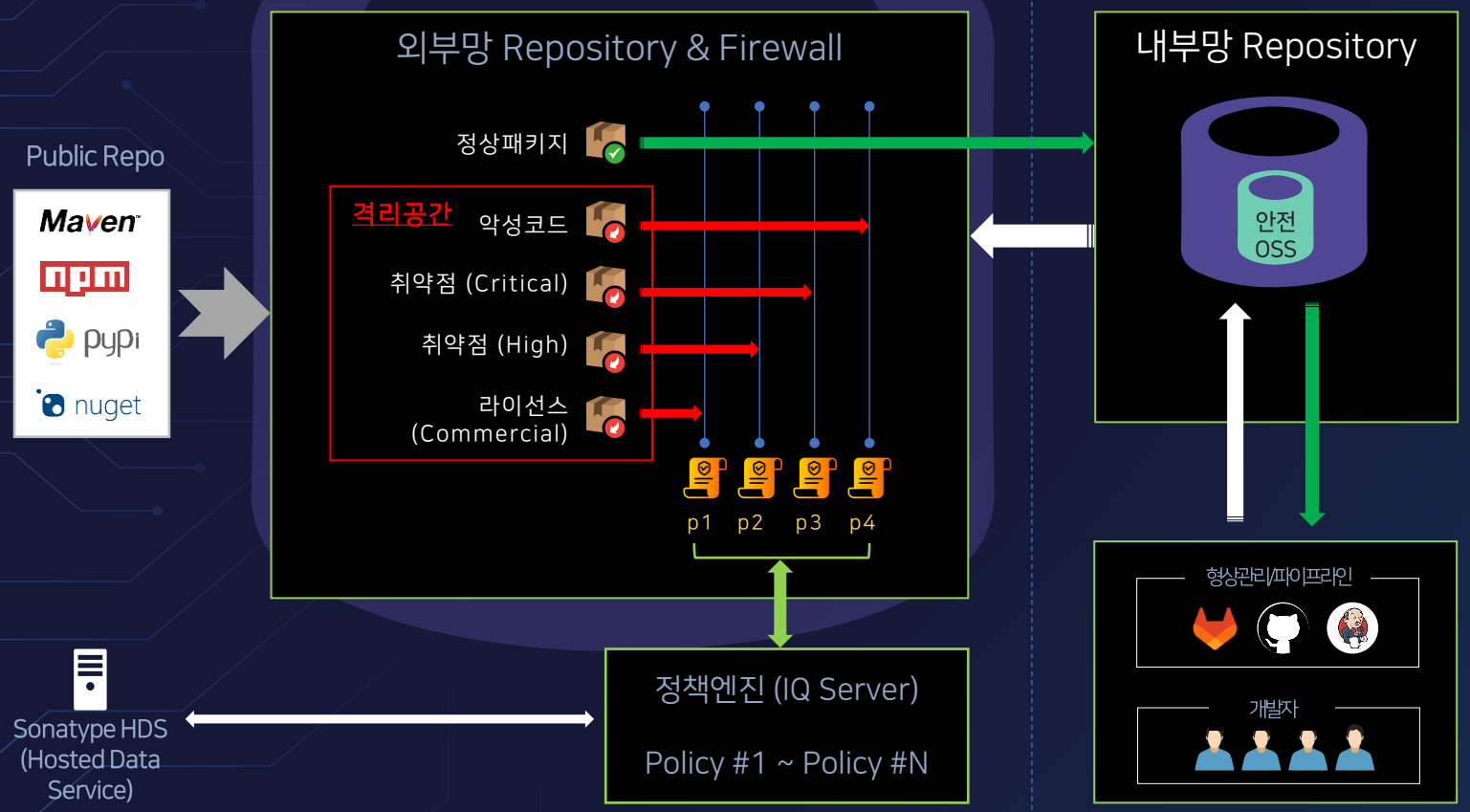
- 정책 기반의 반입 거버넌스
- 전수검사를 통해 오픈소스 리스크 감소

Sonatype Platform 표준 구성 (Repo + FW)



정책설정을 통한 반입통제 자동화

- Nexus상에 FW 기능 탑재로 반입 오픈소스 전수 검사
- 악성코드/멀웨어, 라이선스 및 취약점을 정책 기반으로 통제
- 개발자는 기존 프로세스의 변경 없음
- 자동화된 거버넌스 정책 적용으로 관리비용 절감



Developer Friendly

- 신규 오픈소스를 사용하기 위해 추가적인 프로세스 필요 없음
- 사전 신청 및 승인 과정이 필요 없으므로 개발자 생산성 향상
- 자동화된 반입 프로세스로 개발시간 단축

반입 거버넌스 자동화

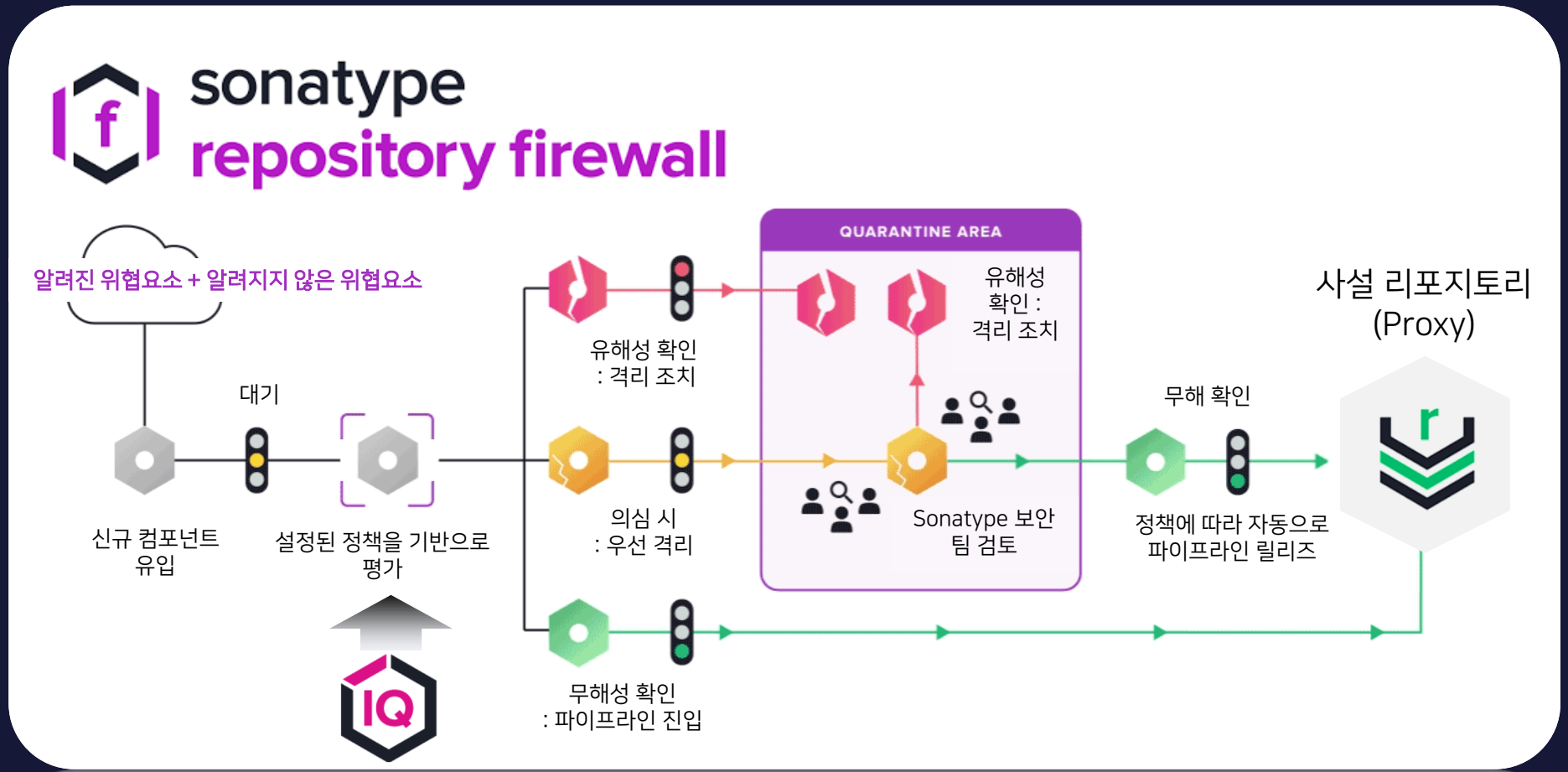
- 전사차원의 일관된 반입 정책 수립
- 사전 정의된 정책을 기반으로 반입 패키지에 대한 전수검사

안전한 오픈소스 사용환경 확보

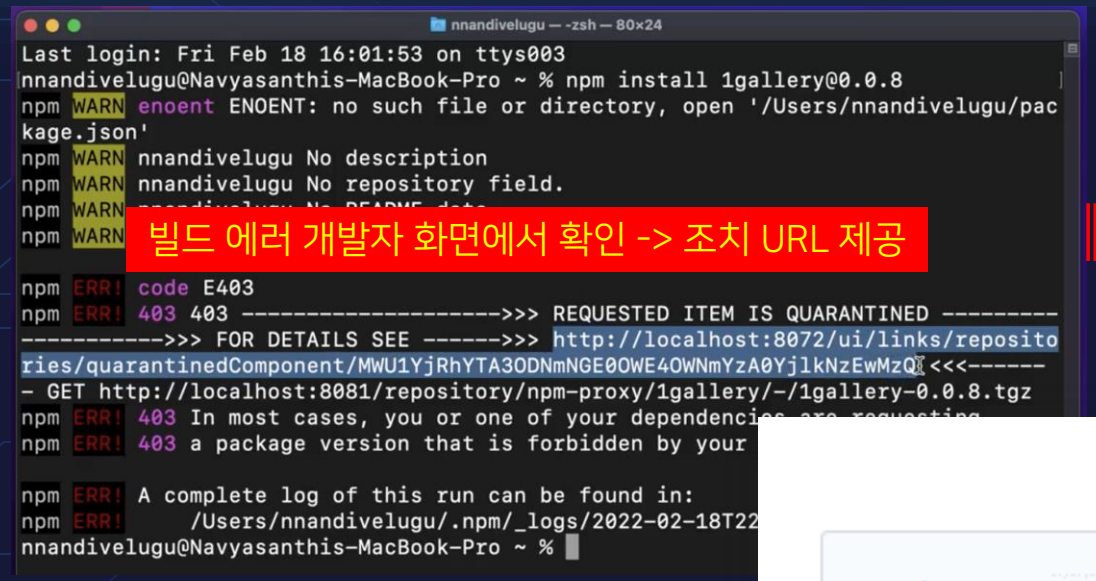
- 글로벌 최대 데이터베이스를 기반으로 악성코드/멀웨어 원천 차단

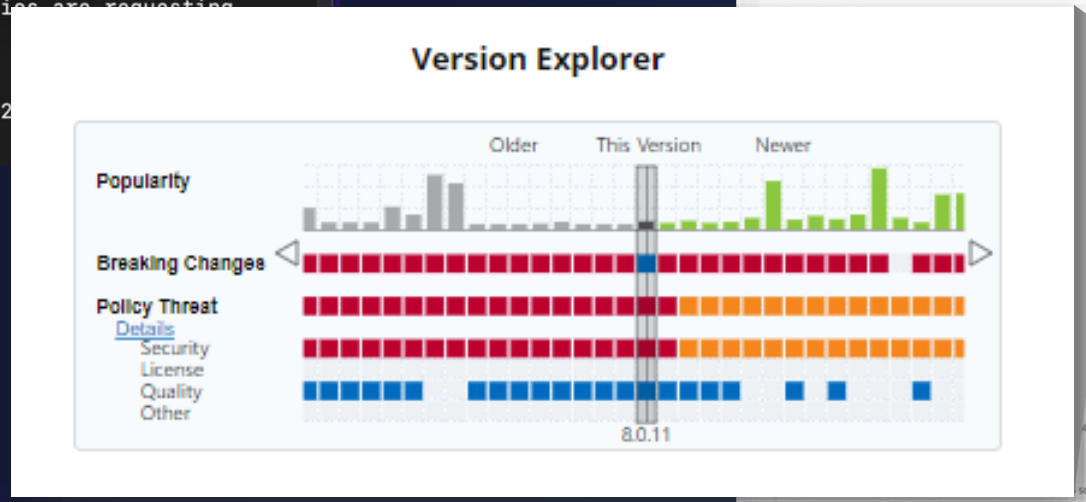
Sonatype Repository Firewall

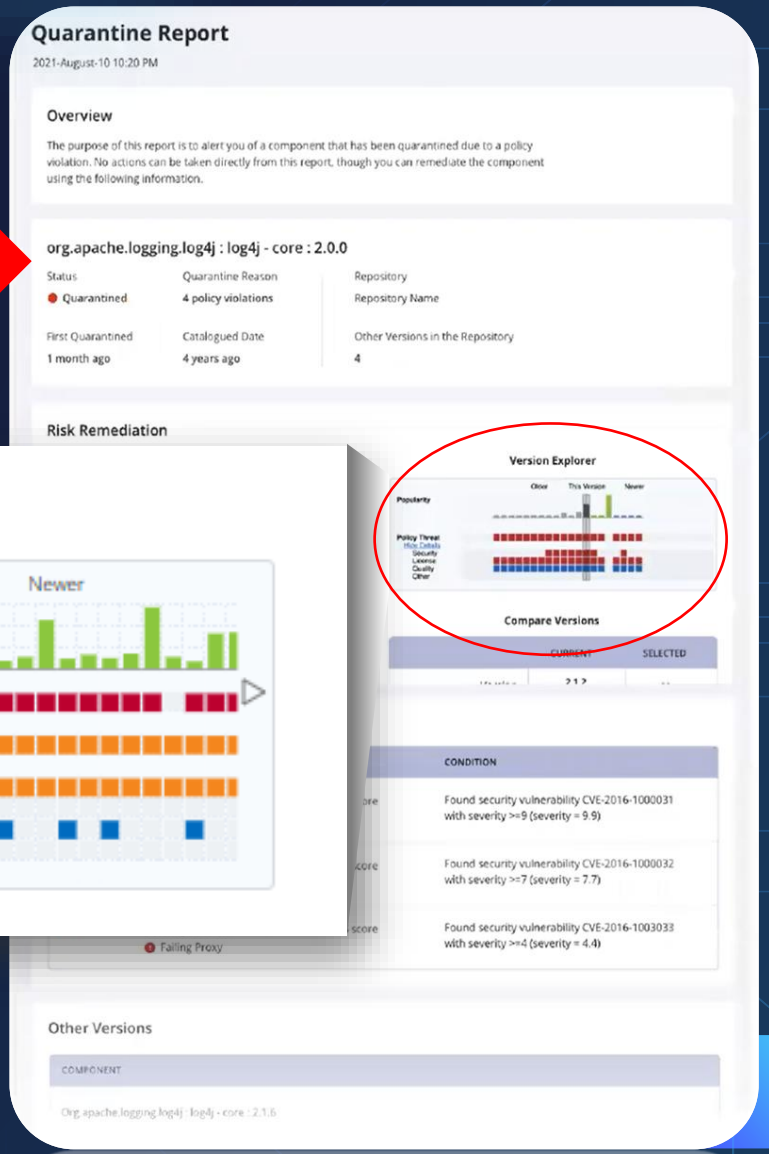
60여개의 시그널을 분석하는 AI/ML 알고리즘을 통해 소프트웨어 공급망을 24X7X365 모니터링 하며 유입되는 오픈소스를 평가하여 유해한 것으로 판단되는 경우 자동으로 다운로드 차단



패키지 차단 시 개발자 조치 : Self Remediation



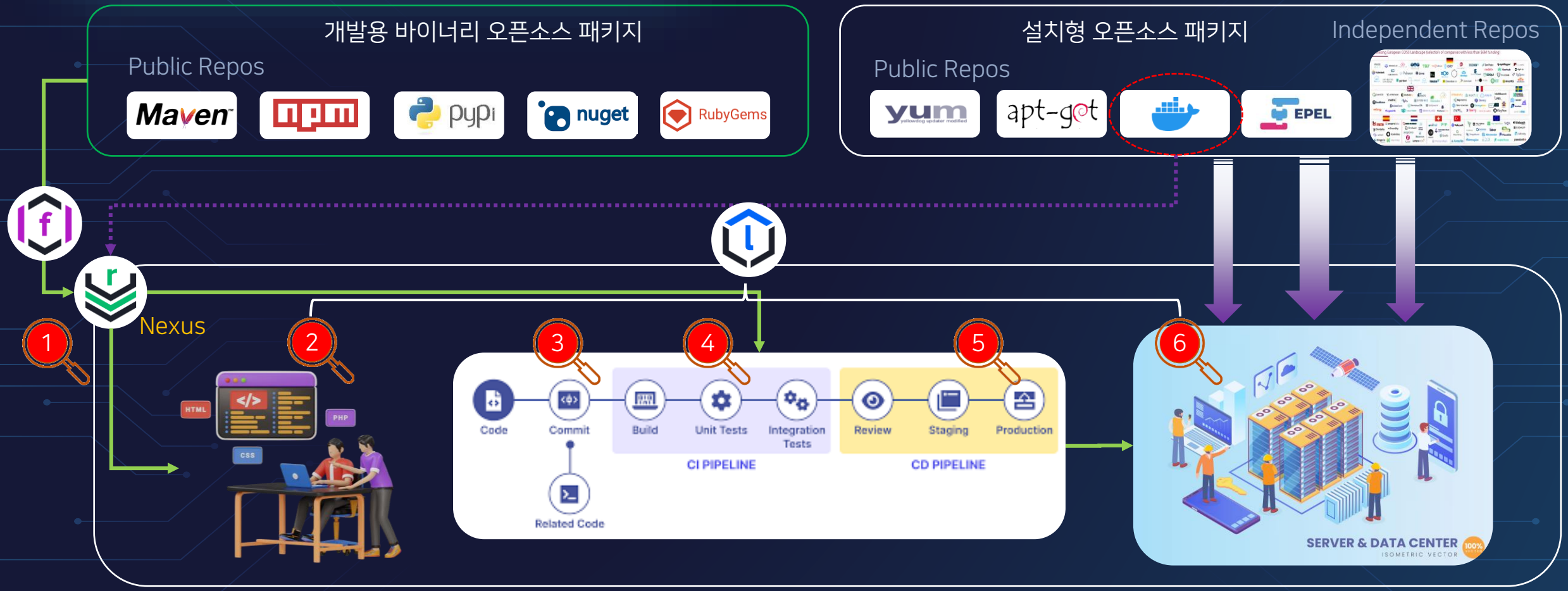




개발자 Self Remediation

1. 어떤 컴포넌트가 차단되었는지?
2. 왜 차단되었는지?
3. 어떻게 하면 차단되지 않는 컴포넌트를 선택할 수 있는지?

오픈소스 유입경로 및 점검



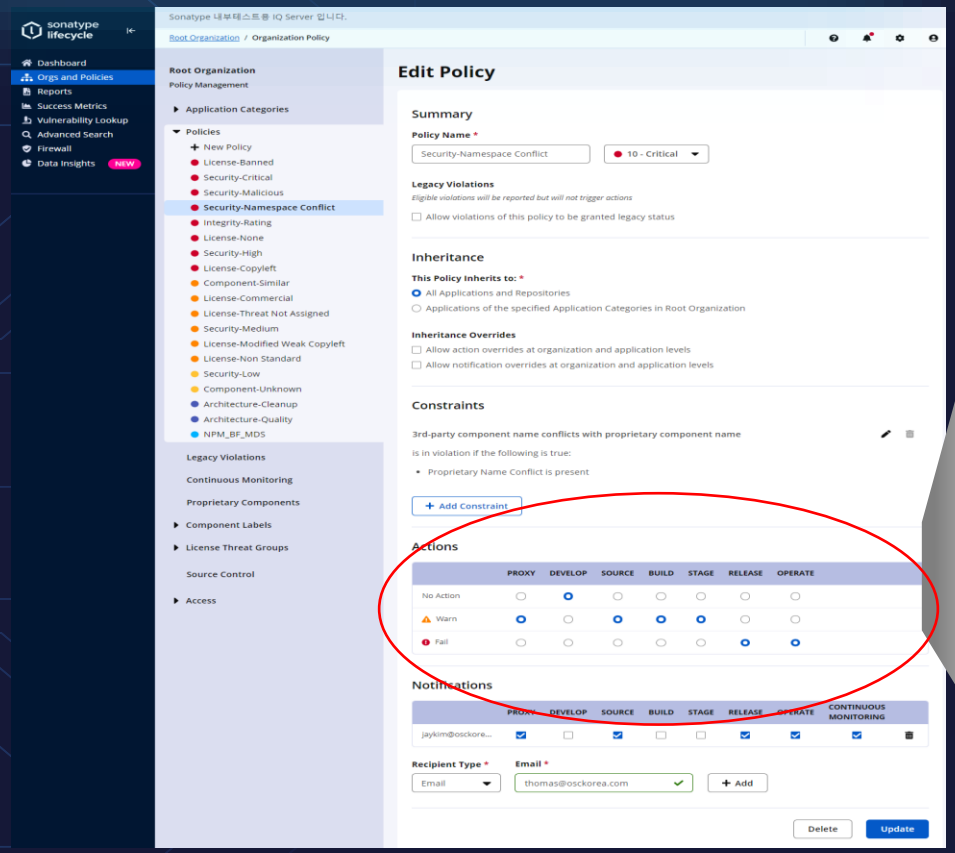
SCA (Software Composition Analysis)

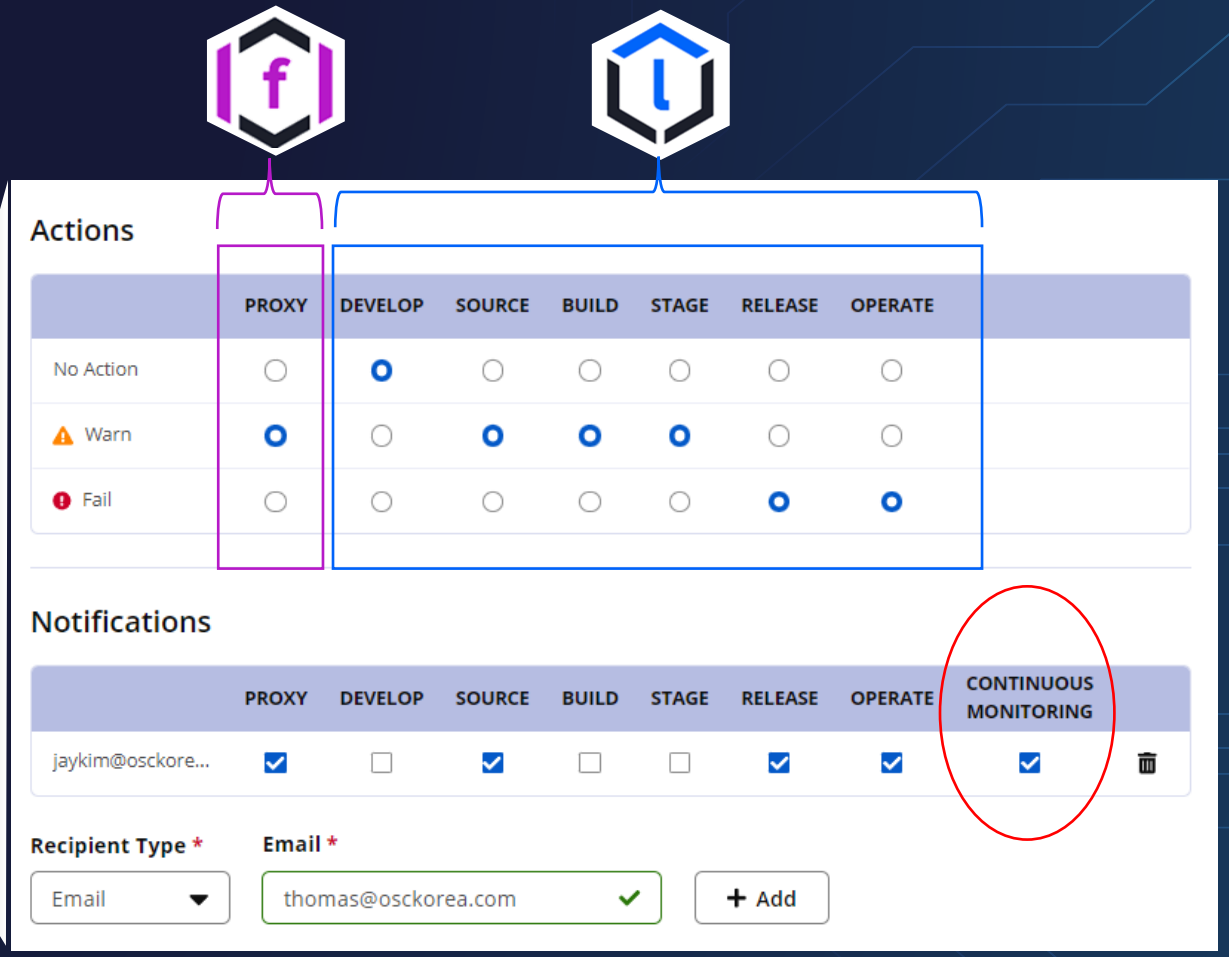
(Sonatype Lifecycle, Blackduck, Mend, Labrado, Sparrow SCA)

Vulnerability Risk Management

(Nessus/Tenable, Rapid7, Qualys)

단계별 거버넌스 정책 적용

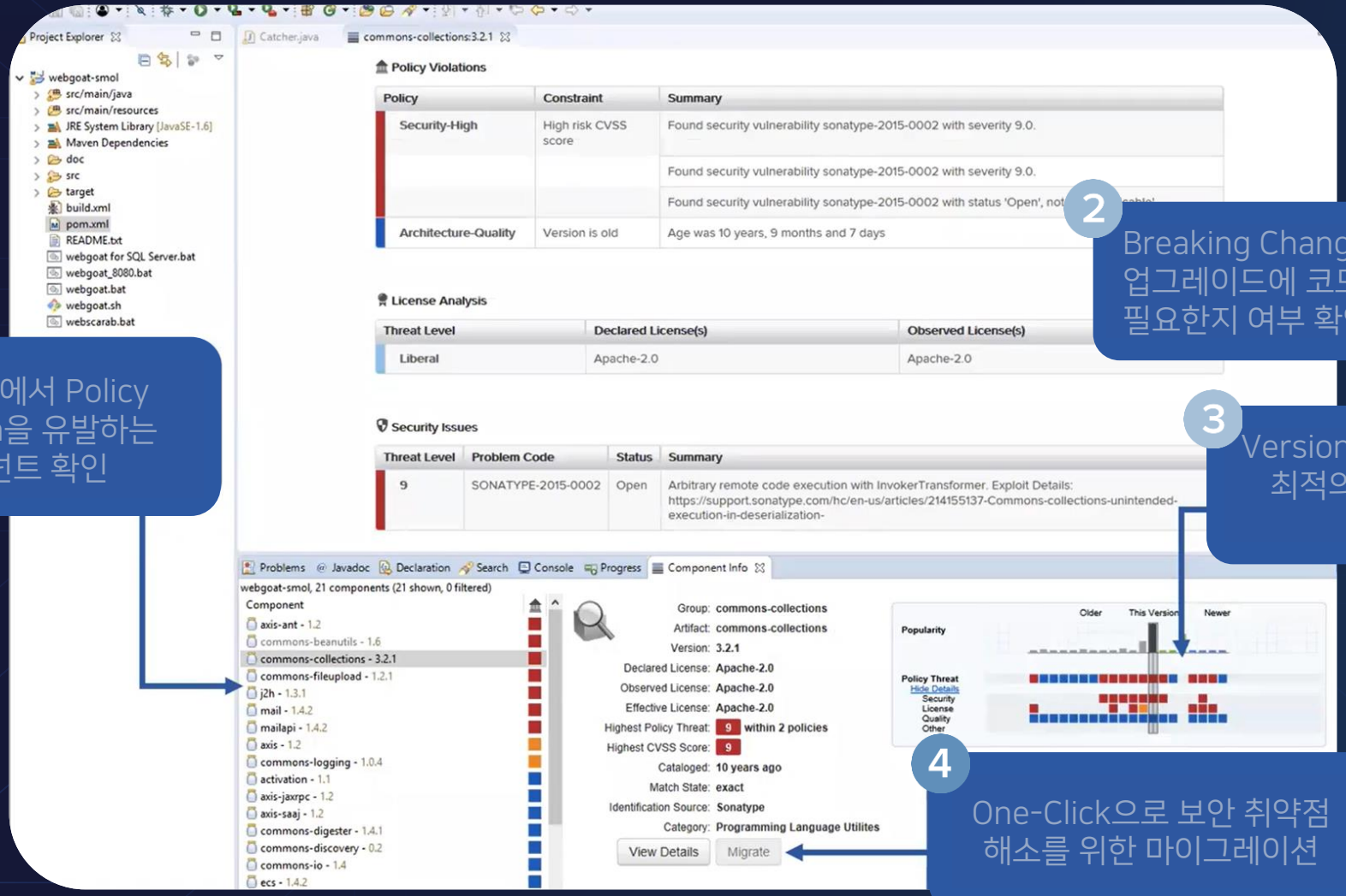




Eclipse · IntelliJ IDEA · VS Code · Github · Gitlab · Bitbucket · Azure DevOps · Jenkins · Bamboo · CircleCI · Github Action · TeamCity · JIRA · Slack

개발자 IDE 환경 지원

Eclipse IntelliJ IDEA VS Code Visual Studio



1 IDE 환경에서 Policy Violation을 유발하는 컴포넌트 확인

2 Breaking Changes : 업그레이드에 코드변경이 필요한지 여부 확인

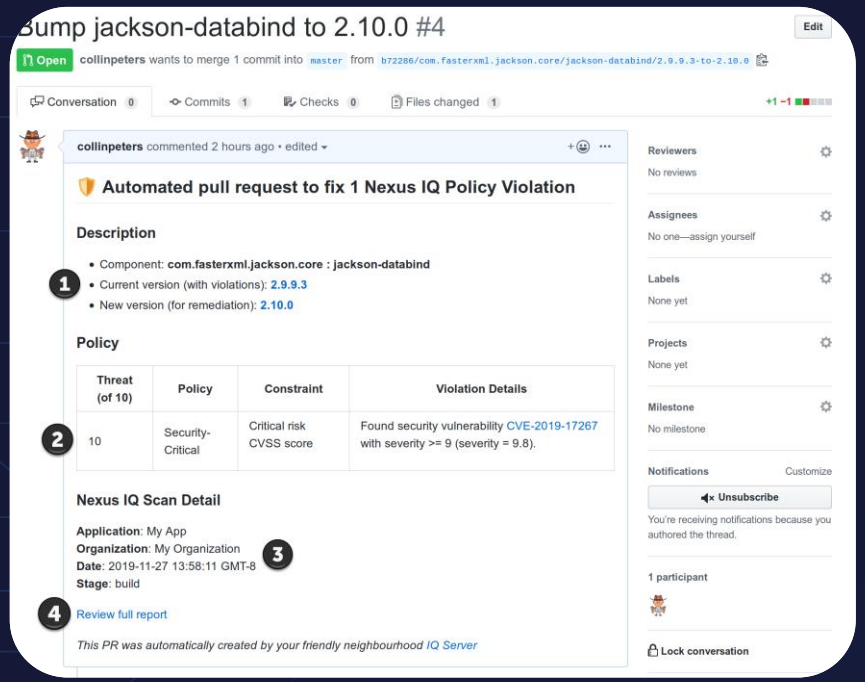
3 Version Explorer를 통해 최적의 컴포넌트 버전 선택

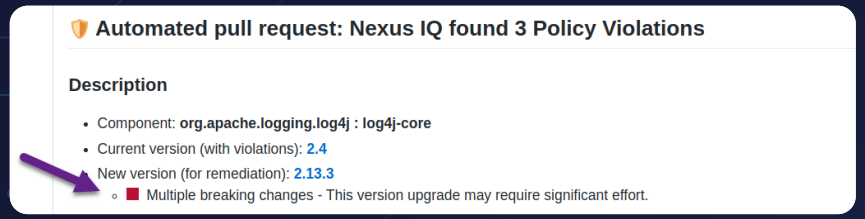
4 One-Click으로 보안 취약점 해소를 위한 마이그레이션

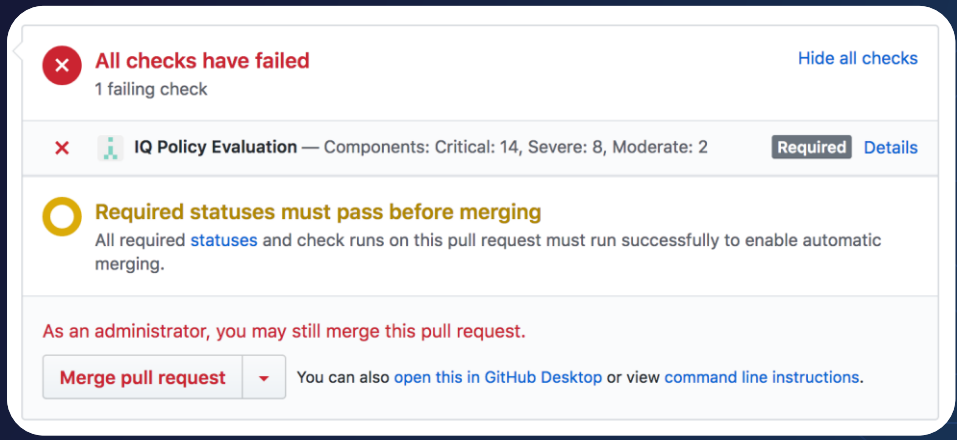
Source Control (형상관리시스템) 지원

- Commit에 대한 자동 피드백, Merge Blocking, 자동 Pull Request
- Breaking Changes 및 Transitive Dependency에 대해 포괄적인 수정 권고안

Azure DevOps Bitbucket Github Gitlab



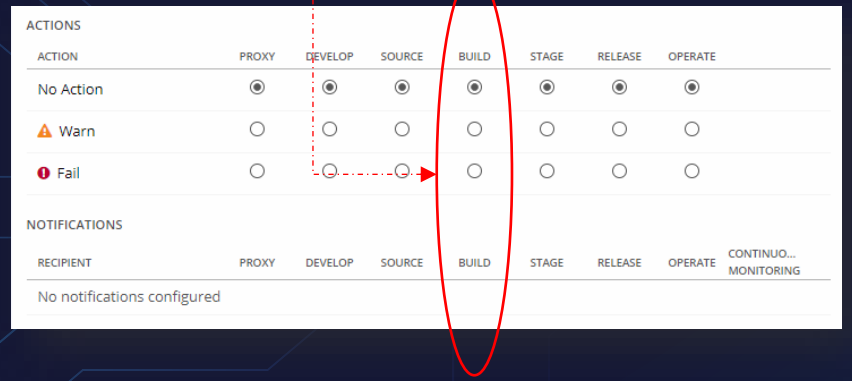




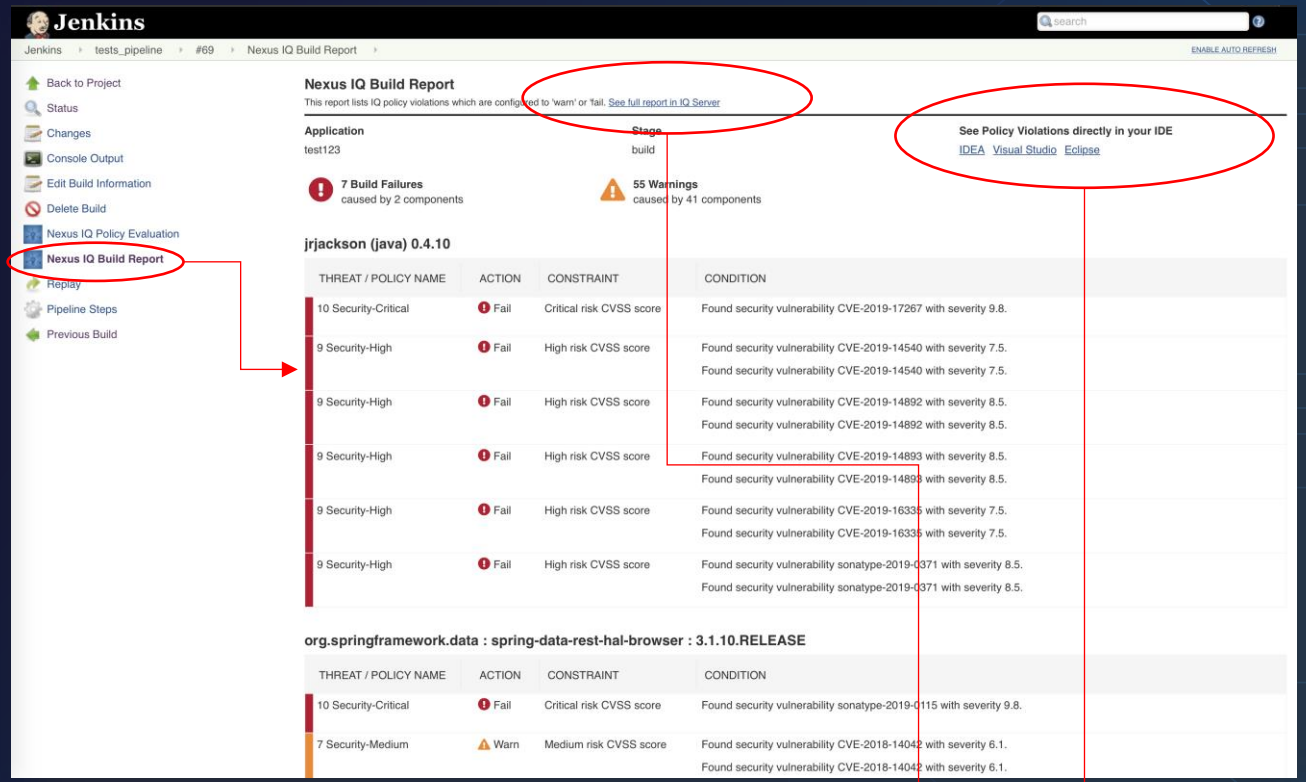
CI Integration (e.g. Jenkins)

Jenkins Script (설정)

```
nexusPolicyEvaluation(
                iqApplication: 'SampApp',
                iqInstanceId: 'MyNexusIQServer1',
                iqScanPatterns: [[scanPattern: '**/*.js'], [scanPattern: '**/*.zip']],
                iqStage: 'build',
                iqOrganization: '55040769ec08424e84049356a3362d07'
)
```



Jenkins Build 리포트



IQ Server를 통한 Full Report 확인

IDE를 통한 Policy Violation 확인

자동화된 오픈소스 관리체계를 위한 세부요건

Shift Left
(시프트 레프트)

오픈소스 유입 관문부터 실시간 통제

nexus repository

repository firewall

개발자 스스로 최적의 컴포넌트 선택

Accuracy
(정확성)

글로벌 최대 데이터베이스 운영

- Maven 운영관리 주체
- NPM/PYPI AI/ML 모니터링
- +65명의 전문 보안연구원
- 전세계 최대 Repo 공급사

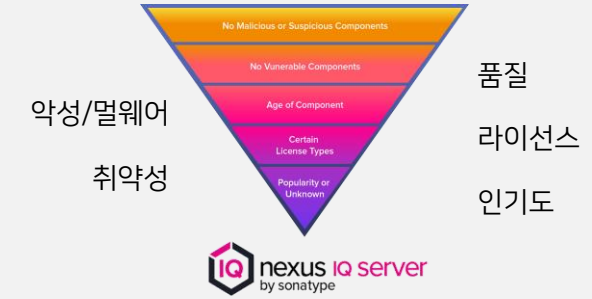
Advanced Binary Fingerprint 식별 기술

```
<item key="013b4d333e95f3a5ac765fc2a3ab05e9f29d7952"
path="ch/qos/logback/core/util/Loader.class"
sha1="6cdbcfa9150af71c7b6b3adfbbc1e1e940f9413e"
sha1JA001="2f9768f33c106400ae23863165643d167a25e8ba"
sha1JB001="878d54d1c132ddeee47ec7ebd9cefbd8b31cb5ac"
sha1JC001="f65040a6798ab66c56ce0ef163195454a68c5921"
sha1JD001="4f093c9bd65a0e6d233171b3362109ab5b372235"/>
```

자동화 필수요건

Flexible Policy
(유연한 정책)

다양한 속성을 통한 정책 정의



단계별 차등화 된 정책 적용

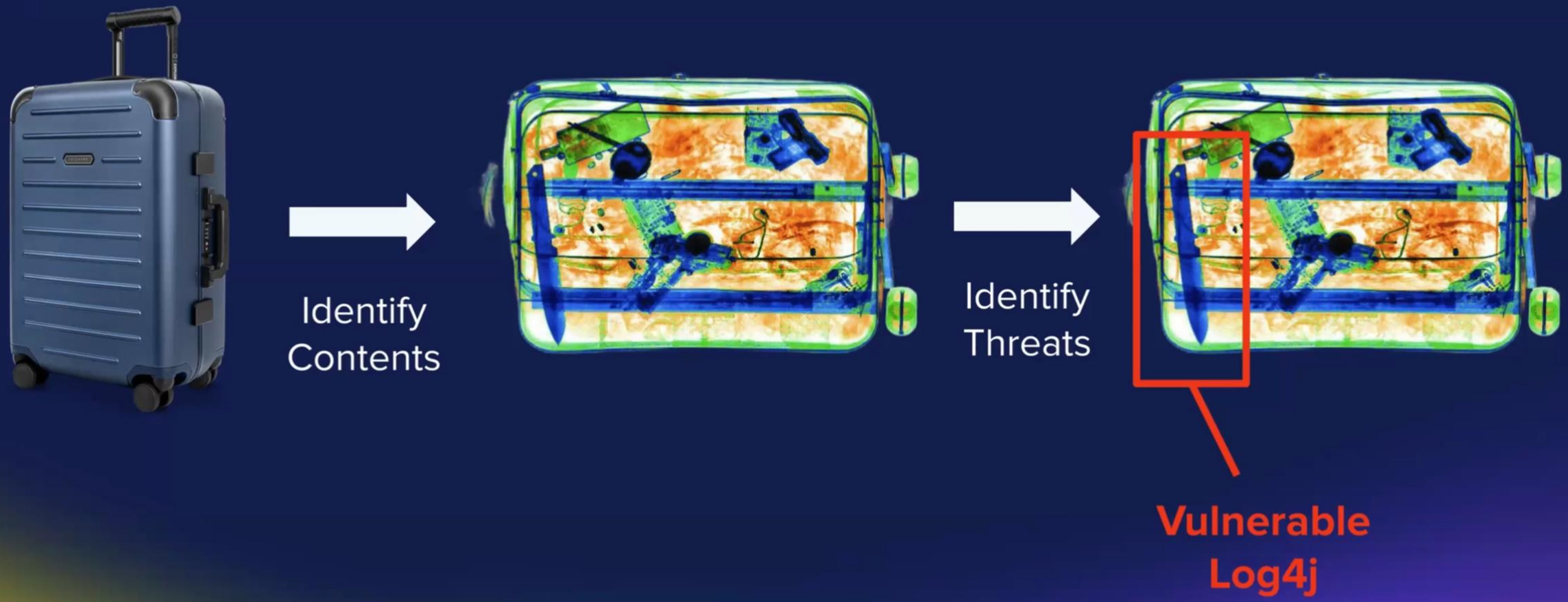
ACTIONS

ACTION	PROXY	DEVELOP	SOURCE	BUILD	STAGE	RELEASE	OPERATE
No Action	○	◉	◉	◉	◉	◉	◉
⚠ Warn	○	○	○	○	○	○	○
❗ Fail	◉ ❗	○	○	○	○	○	○

Beyond SDLC
SBOM Manager

- 소프트웨어 공급망 관리의 도전과제 및 공격 기법
- 오픈소스 거버넌스 자동화 고려사항
- 오픈소스 거버넌스 자동화 구현 방안
- SBOM 통합관리를 위한 SBOM Manager

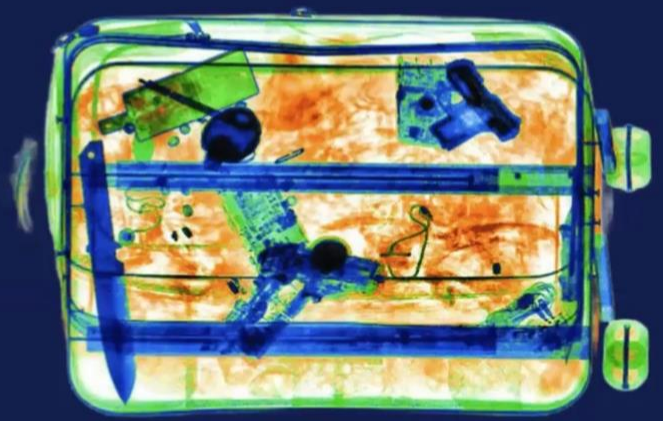
Software Composition Analysis



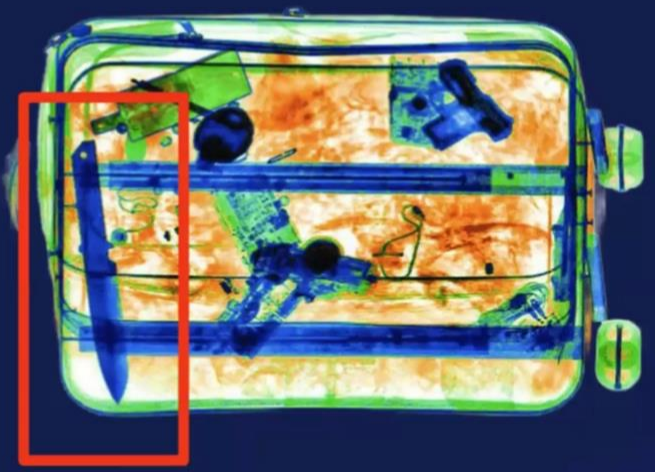
sonatype

Software Composition Analysis

Identify Contents



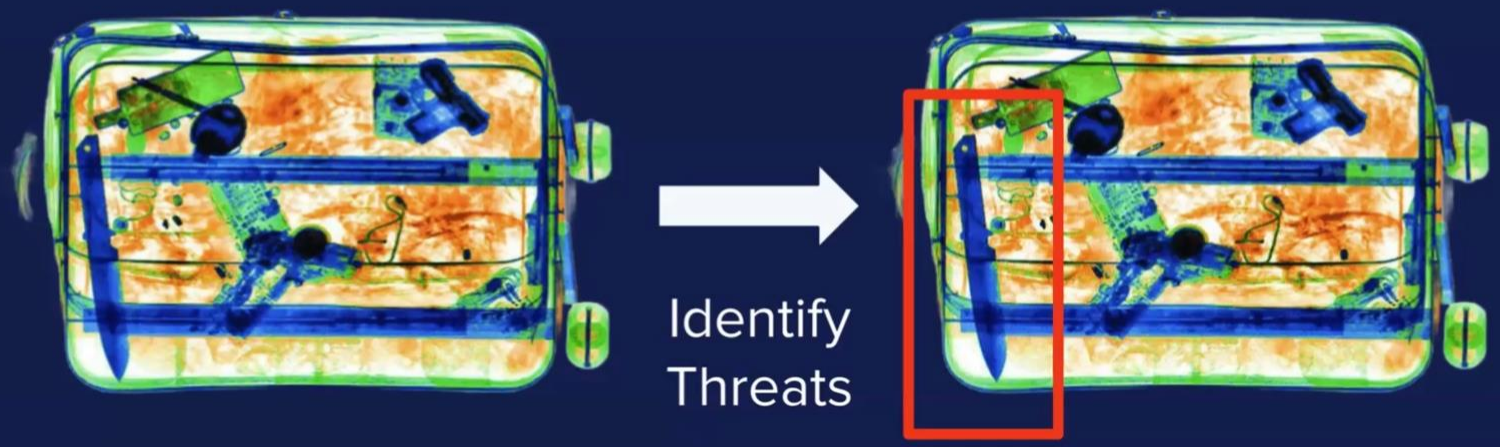
Identify Threats



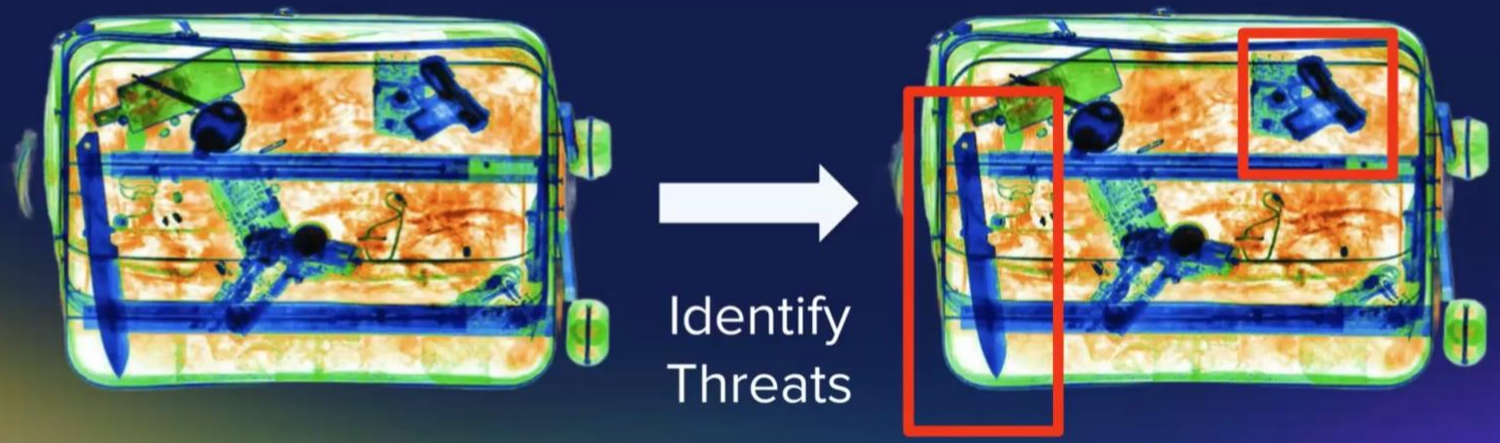
Software Bill of Materials (SBOM) makes this shareable

sonatype

Checking Your Work



Internal Checks



External Checks

sonatype

Regulations: You Need SBOM Manager If...

신용카드 사용자 데이터를 저장,처리 및 전송하는 경우
(PCI 4.0)

EU에 소프트웨어가 포함된 유형의 제품을 판매하는 경우
(CRA)

EU에서 디지털서비스를 운영하거나 중요한 산업을 지원하는 경우
(NIS2)

금융기관이거나 금융조직을 지원하는 경우
(DORA)

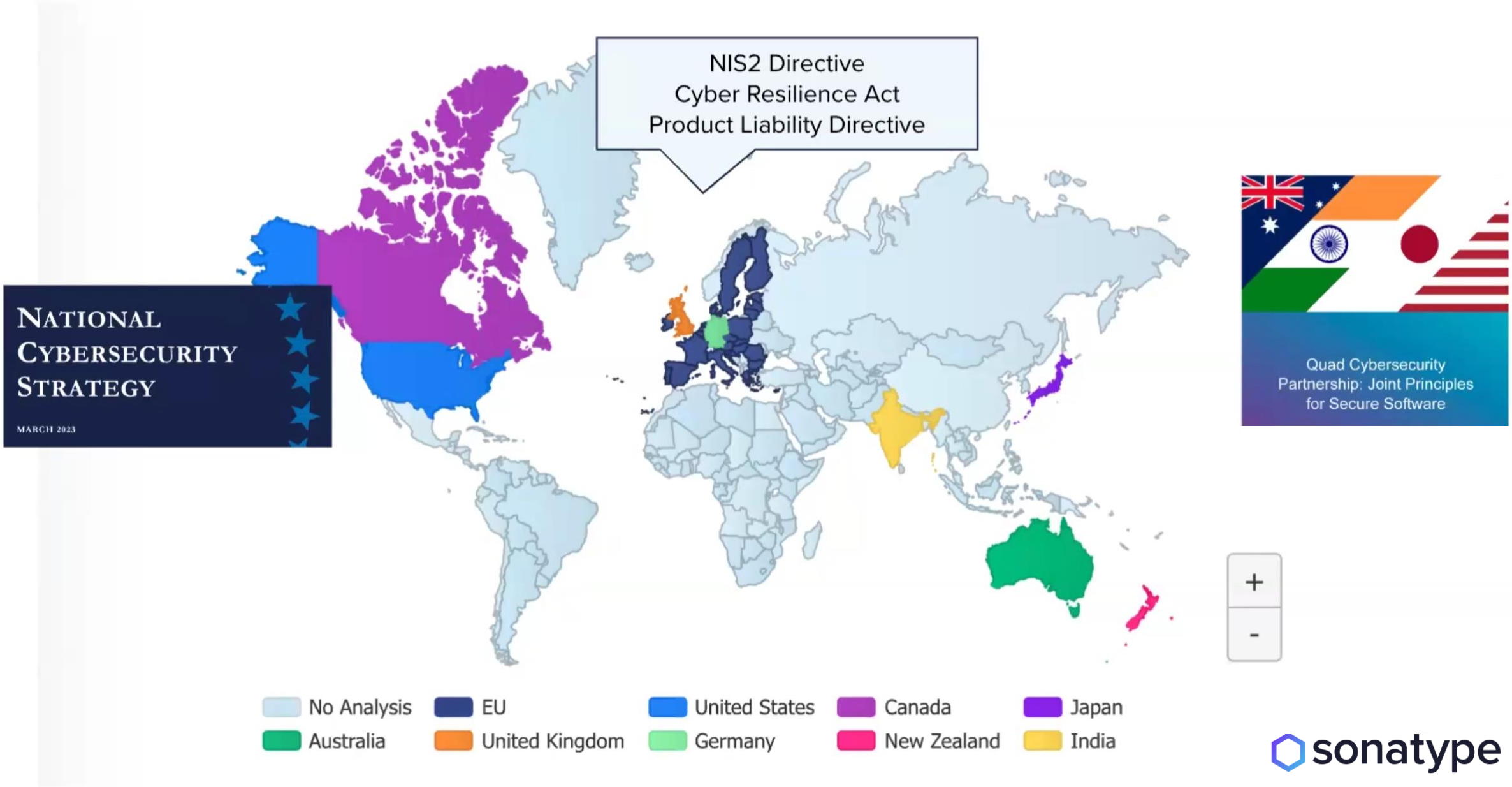
의료장치를 판매하는 경우
(FD&C Act / FDA)

미국 연방정부에 소프트웨어를 판매하는 경우
(CISA Attestation Form)

미국 연방정부를 위한 계약 건으로 소프트웨어를 개발하는 경우
(FAR)

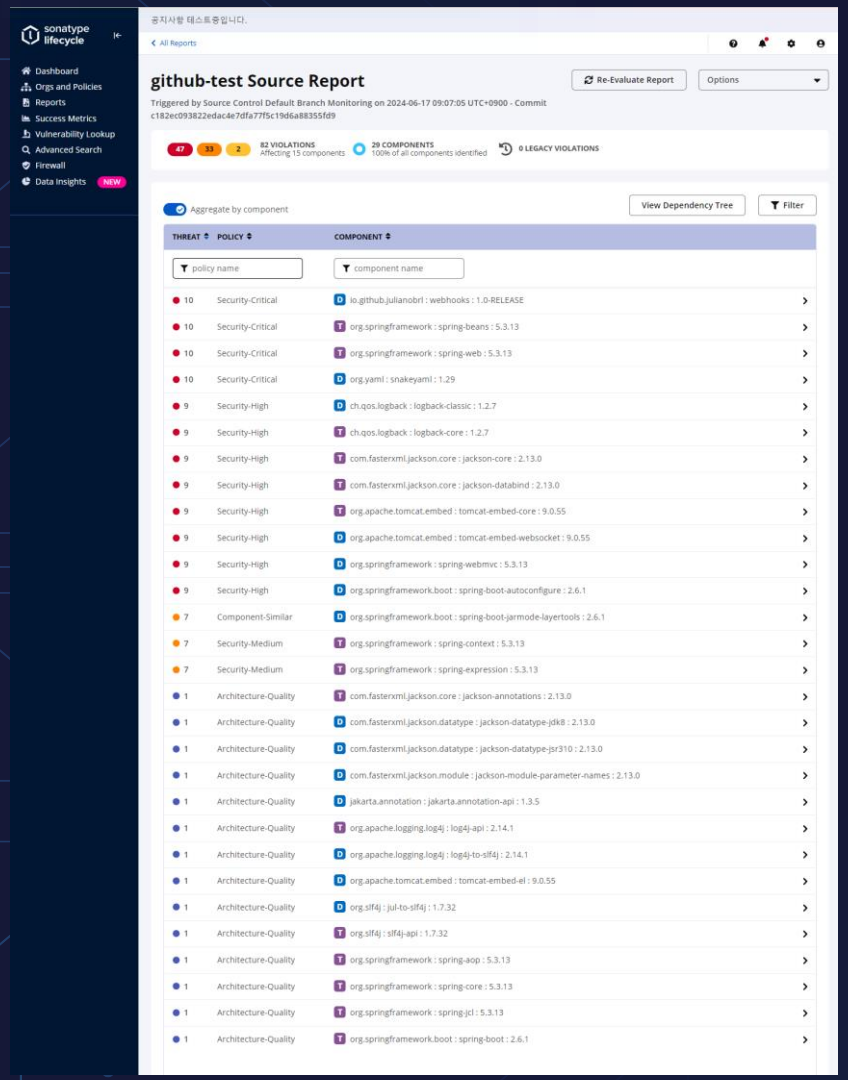
sonatype

Regulations Are ~~Coming~~ Here

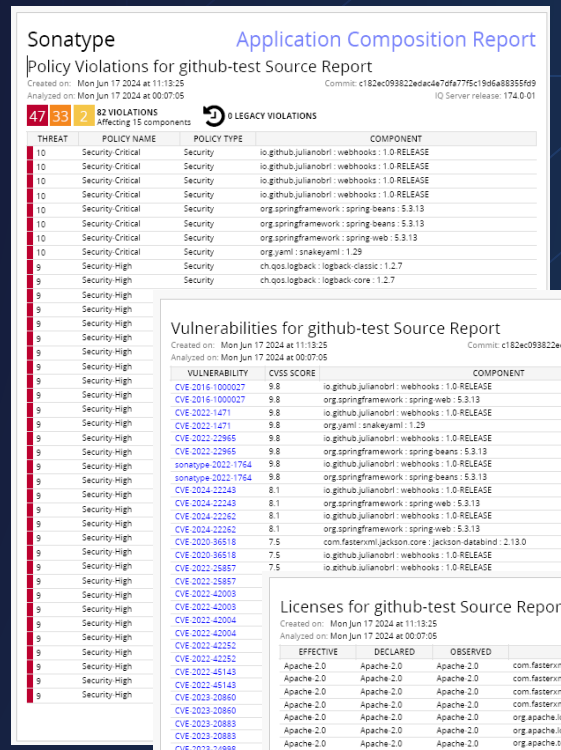


sonatype

SBOM 지원


sonatype lifecycle
Dashboard
Orgs and Policies
Reports
Success Metrics
Vulnerability Lookup
Advanced Search
Firewall
Data Insights
github-test Source Report
Re-Evaluate Report
Options
82 VIOLATIONS
29 COMPONENTS
0 LEGACY VIOLATIONS
View Dependency Tree
Filter


Dependency Tree
Expand All
Collapse All

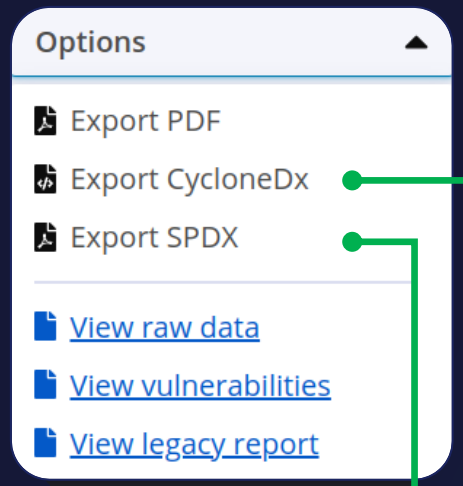

Sonatype
Application Composition Report
Policy Violations for github-test Source Report
82 VIOLATIONS
0 LEGACY VIOLATIONS

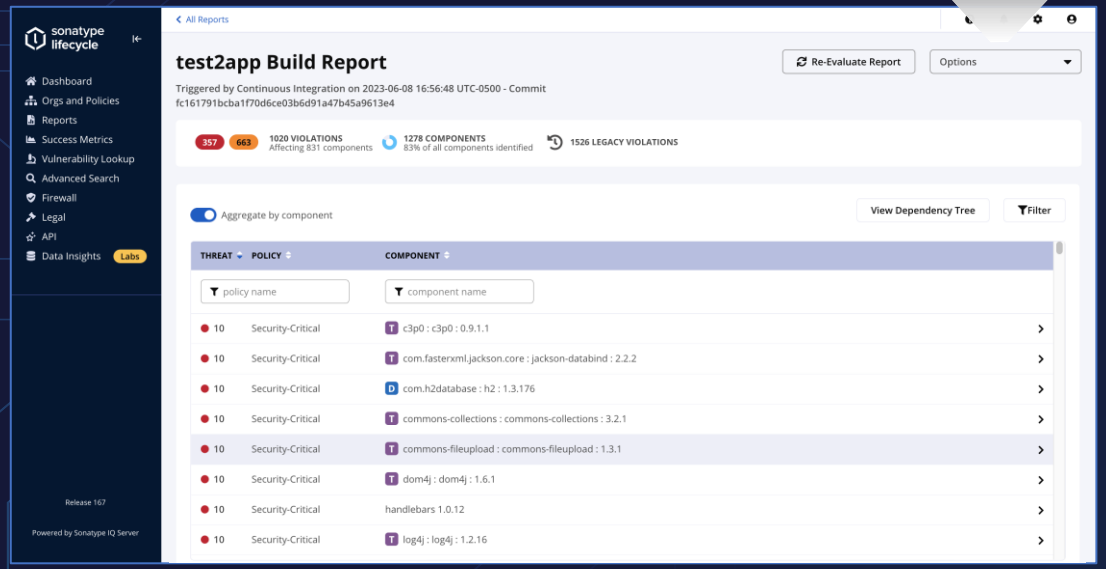

Vulnerabilities for github-test Source Report

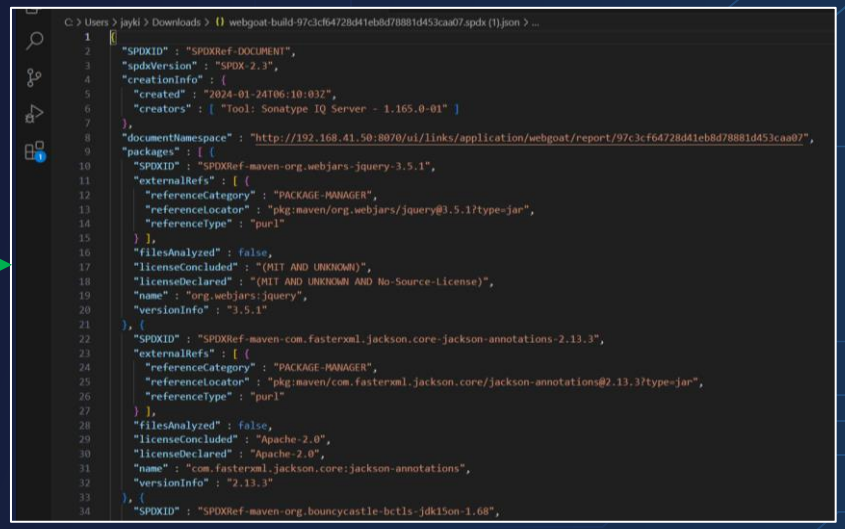
Licenses for github-test Source Report

표준 SBOM

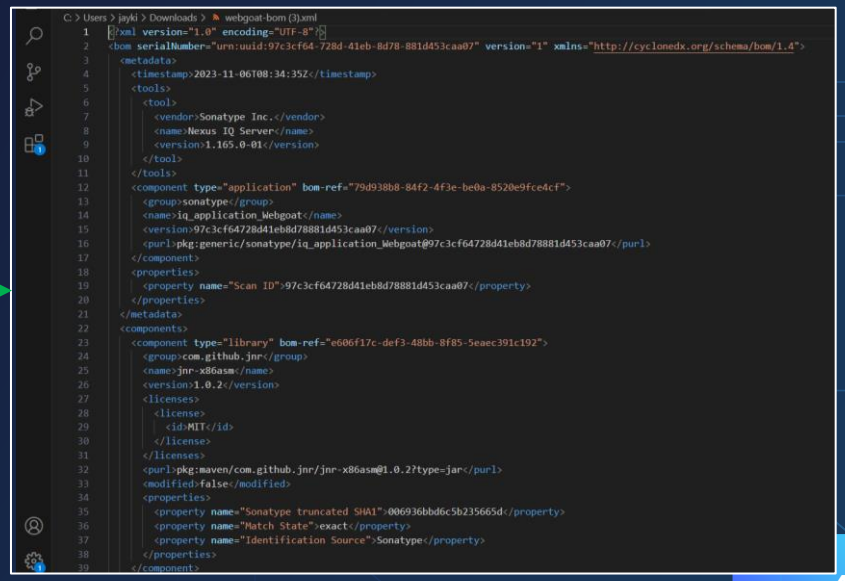
Application 분석 Report는
CycloneDX 또는 SPDX 포맷의
SBOM 형태로 생성할 수 있습니다.





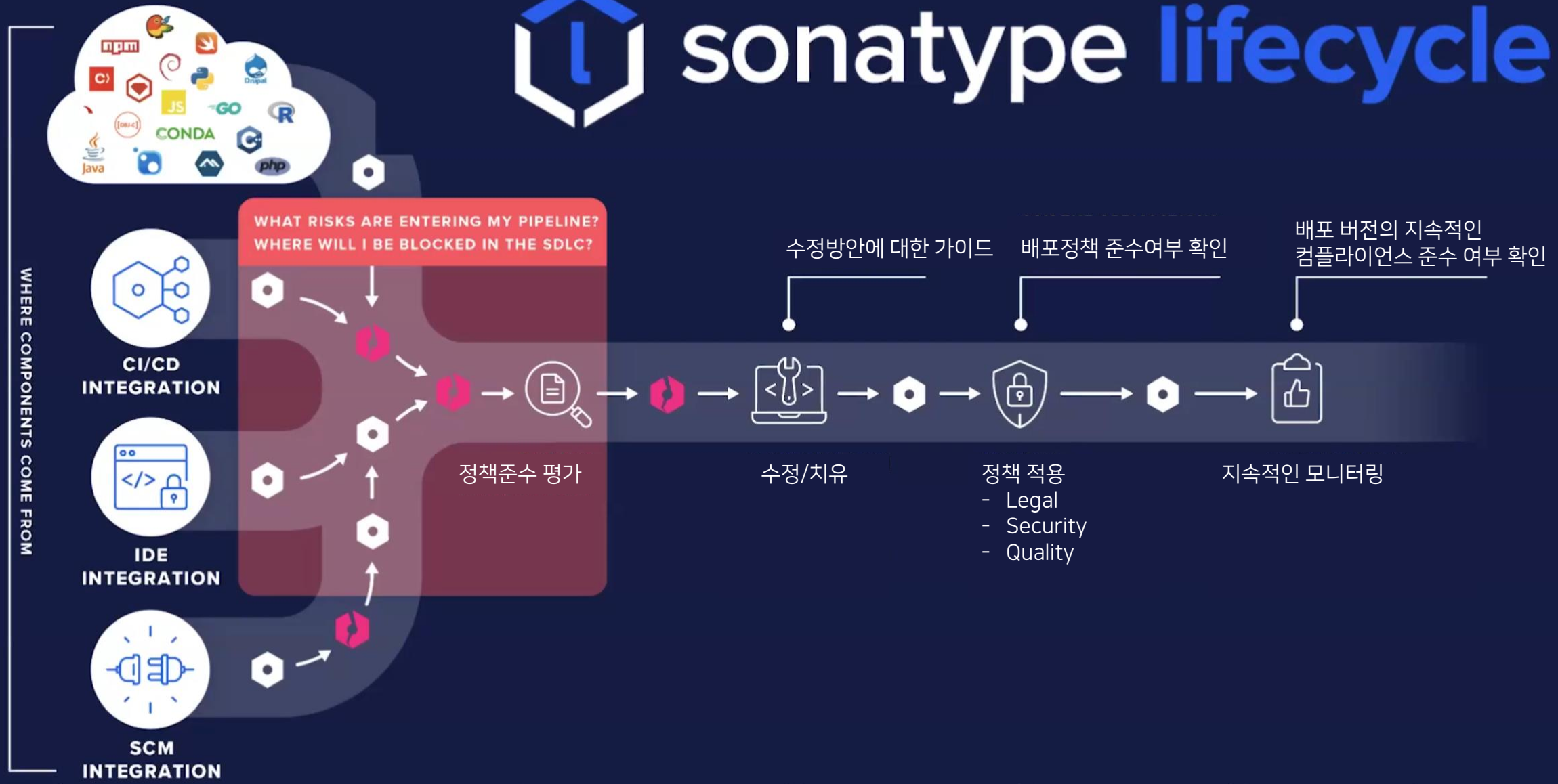


CycloneDX (bom.xml)

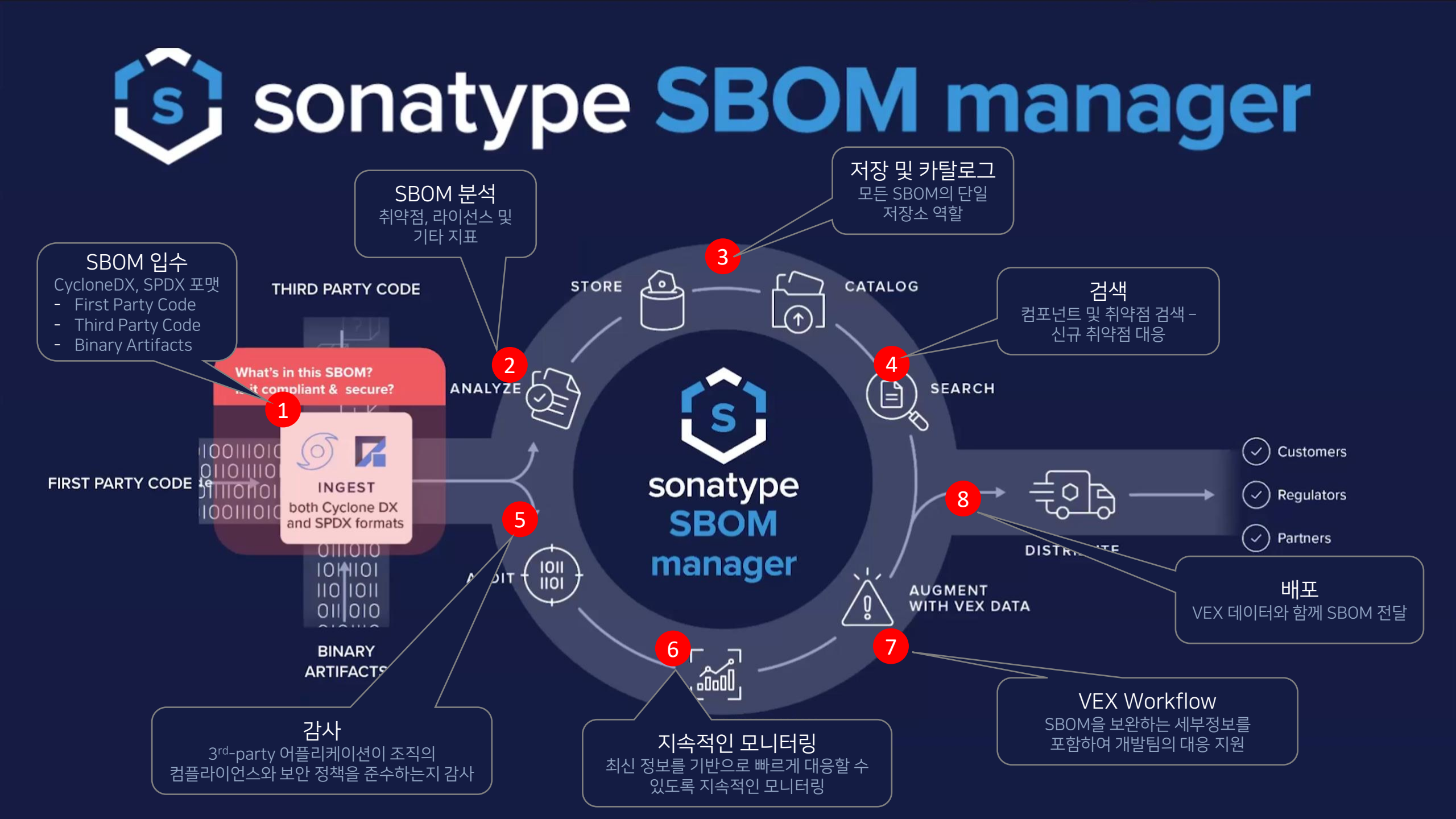


SPDX (bom.json)

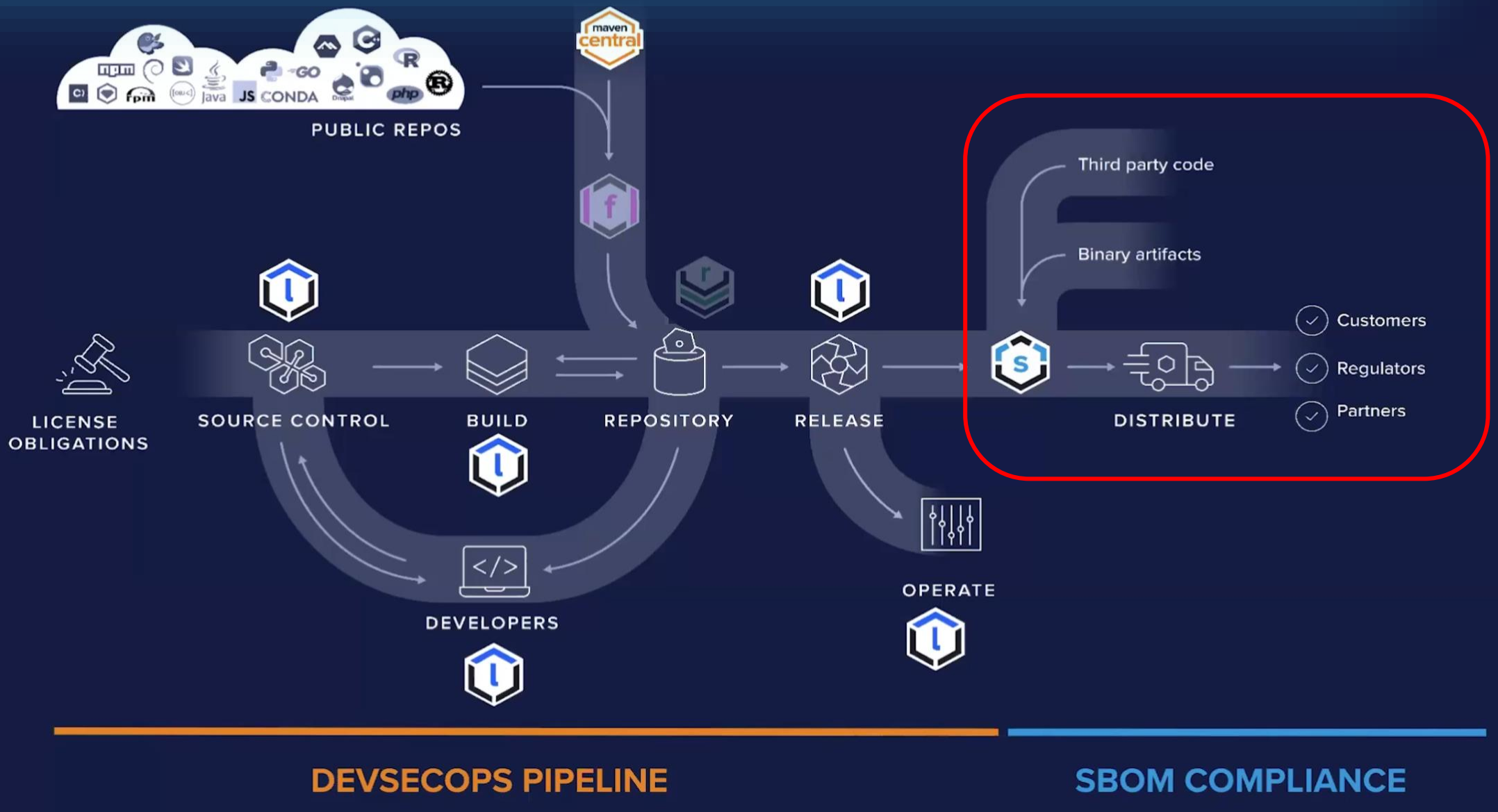
sonatype lifecycle



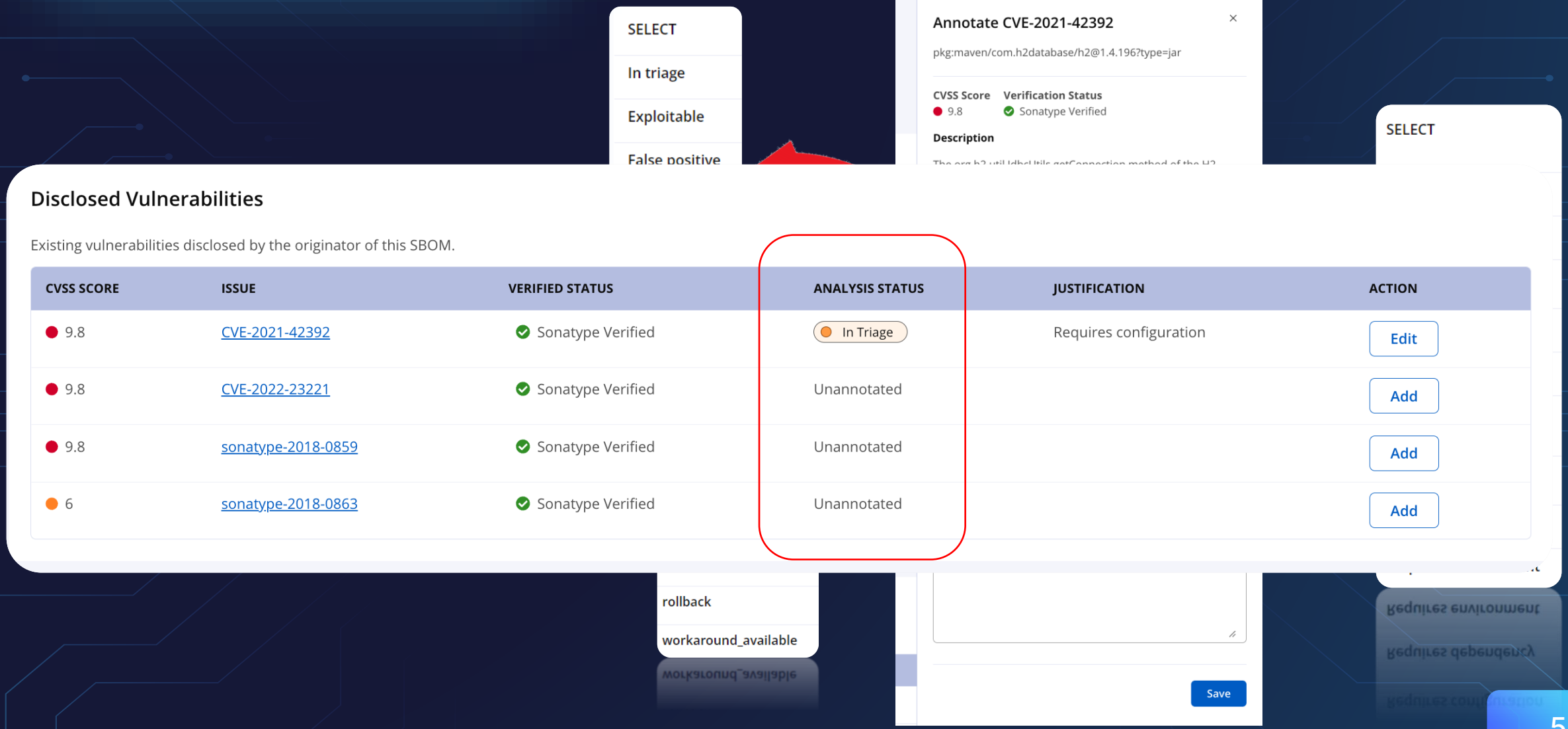
sonatype SBOM manager



The essential duo of SCA and SBOM management



VEX Annotations



Sonatype SBOM Manager vs. SCA

Software Composition Analysis	SBOM Management
Part of the SDLC 개발 파이프라인에서 Legal, security, Quality 리스크가 없는 최적의 컴포넌트를 선택하기 위해 활용	**Beyond the SDLC** 배포된 소프트웨어를 관리하고, 필요시 패치를 반영하거나 요청할 수 있도록 소프트웨어 개발 생명주기 이후 버전별로 관리
최종 버전 관리	배포된 모든 버전에 대한 관리
Development (SDLC) 중심	Release / Deployment 중심
Remediation 중심	Compliance 중심
주사용자 : 개발팀, 보안팀, DevOps팀	주사용자 : 컴플라이언스, 구매팀, 및 보안팀

sonatype

감사합니다

OSC Korea